Alcatel-Lucent OpenTouch™ Session Border Controller

User Manual
Ver. 6.4

8AL90524USAA ed01

Table of Contents

1 Introduction 10

2 Installing the Software 12

3 Changing Default IP Address to Suit your Network Addressing Scheme ... 13

4 Configuring Advanced Network Settings using the Web Interface 15

5 Web-Based Management 17

5.1 Getting Acquainted with the Web Interface 17
5.1.1 Computer Requirements 17
5.1.2 Accessing the Web Interface 18
5.1.3 Areas of the GUI 19
5.1.4 Toolbar Description 20
5.1.5 Navigation Tree 20
5.1.5.1 Displaying Navigation Tree in Basic and Full View 21
5.1.5.2 Showing / Hiding the Navigation Pane 23
5.1.6 Working with Configuration Pages 24
5.1.6.1 Accessing Pages 24
5.1.6.2 Viewing Parameters 25
5.1.6.3 Modifying and Saving Parameters 26
5.1.6.4 Entering Phone Numbers 27
5.1.6.5 Working with Tables 28
5.1.7 Searching for Configuration Parameters 31
5.1.8 Creating a Login Welcome Message 33
5.1.9 Getting Help 34
5.1.10 Logging Off the Web Interface 35
5.2 Using the Home Page 36
5.3 Configuring Web User Accounts 38
5.4 Configuring Web Security Settings 41
5.5 Web Login Authentication using Smart Cards 42
5.6 Configuring Web and Telnet Access List 43
5.7 Configuring RADIUS Settings 45

6 CLI-Based Management 46

6.1 Configuring Telnet and SSH Settings 46

7 SNMP-Based Management 47

7.1 Configuring SNMP Community Strings 47
7.2 Configuring SNMP Trap Destinations 49
7.3 Configuring SNMP Trusted Managers 50
7.4 Configuring SNMP V3 Users 51

8 INI File-Based Management 53

8.1 INI File Format 53
8.1.1 Configuring Individual ini File Parameters 53
8.1.2 Configuring ini File Table Parameters 54
8.1.3 General ini File Formatting Rules 55
8.2 Modifying an ini File 56
8.3 Secured Encoded ini File 56

9 Configuring Certificates 58

9.1 Replacing Device Certificate 58

9.2 Loading a Private Key 61
9.3 Mutual TLS Authentication 62
9.4 Self-Signed Certificates 63

10 Date and Time 64

10.1 Manual Date and Time 64
10.2 Automatic Date and Time through SNTP Server 65

11 Network 68

11.1 Configuring Physical Ethernet Ports 68
11.2 Configuring IP Interface Settings 69
11.2.1 Network Configuration Overview 73
11.2.1.1 Multiple Network Interfaces and VLANs 73
11.2.1.2 Setting Up VoIP Networking 81
11.3 Configuring the IP Routing Table 86
11.3.1 Routing Table Columns 87
11.3.1.1 Destination Column 87
11.3.1.2 Prefix Length Column 87
11.3.1.3 Gateway Column 88
11.3.1.4 Interface Column 88
11.3.1.5 Metric Column 88
11.3.1.6 State Column 88
11.3.2 Routing Table Configuration Summary and Guidelines 89
11.3.3 Troubleshooting the Routing Table 89
11.4 Configuring QoS Settings 90
11.5 DNS 91
11.5.1 Configuring the Internal DNS Table 91
11.5.2 Configuring the Internal SRV Table 92
11.6 NAT (Network Address Translation) Support 92
11.6.1 First Incoming Packet Mechanism 93
11.6.2 No-Op Packets 93
11.7 Configuring NFS Settings 95
11.8 Robust Receipt of Media Streams 97
11.9 Multiple Routers Support 97

12 Security 98

12.1 Configuring Firewall Settings 98
12.2 Configuring General Security Settings 102

13 Media 103

13.1 Configuring RTP/RTCP Settings 103
13.1.1 Configuring RTP Base UDP Port 104
13.1.1.1 Configuring RTP Base UDP Port using Web Interface 104
13.2 Configuring General Media Settings 105
13.3 Configuring Media Realms 105
13.4 Configuring Media Security 108

14 Enabling Applications 109

15 Control Network 110

15.1 Configuring SRD Table 110
15.2 Configuring SIP Interface Table 112
15.3 Configuring IP Groups 114

15.4 Configuring Proxy Sets Table 120
15.5 Configuring NAT Translation per IP Interface 126
15.6 Multiple SIP Signaling and Media Interfaces using SRDs 127

16 SIP Definitions 130

16.1 Configuring SIP General Parameters 130
16.2 Configuring Advanced Parameters 131
16.3 Configuring Account Table 132
16.4 Configuring Proxy and Registration Parameters 135

17 Profiles 137

17.1 Configuring IP Profiles 137

18 Session Border Controller 139

18.1 SBC Overview 139
18.1.1 Overview 139
18.1.1.1 NAT Traversal 140
18.1.1.2 VoIP Firewall 140
18.1.1.3 Topology Hiding 140
18.1.1.4 SIP Normalization 141
18.1.1.5 Survivability 141
18.1.2 SIP Network Definitions 141
18.1.3 SIP Dialog Initiation Process 141
18.1.3.1 Determining Source and Destination URL 142
18.1.3.2 Source IP Group Classification 143
18.1.3.3 SBC IP-to-IP Routing 145
18.1.3.4 IP-to-IP Inbound and Outbound Manipulation 145
18.1.3.5 SIP Header Manipulation 148
18.1.4 User Registration and Internal Database 150
18.1.4.1 Initial Registration Request Processing 150
18.1.4.2 Internal Database 151
18.1.4.3 Routing using Internal Database 152
18.1.4.4 Registration Refreshes 152
18.1.4.5 Registration Restriction Control 152
18.1.5 SBC Media Handling 154
18.1.5.1 Media Anchoring without Transcoding (Transparent) 155
18.1.5.2 No Media Anchoring 156
18.1.5.3 Interworking DTMF Methods 157
18.1.5.4 Restricting Coders 157
18.1.5.5 SRTP-RTP Transcoding 160
18.1.5.6 Multiple RTP Media Streams per Call Session 160
18.1.6 Fax Negotiation and Transcoding 161
18.1.7 SIP Dialog Admission Control 161
18.1.8 Limiting SBC Call Duration 162
18.1.9 SIP Authentication Server for SBC Users 162
18.1.10 Handling SIP 3xx Redirect Responses 162
18.1.11 Interworking SIP Diversion and History-Info Headers 164
18.1.12 Call Survivability 165
18.1.12.1 Auto-Provisioning of Subscriber-Specific Information for BroadWorks Server for Survivability 165
18.1.12.2 BroadSoft's Shared Phone Line Call Appearance for SBC Survivability 166
18.1.12.3 Call Survivability for Call Centers 167
18.1.12.4 Survivability Mode Display on Aastra IP Phones 169
18.1.13 Call Forking 171
18.1.13.1 Initiating SIP Call Forking 171
18.1.13.2 SIP Forking Initiated by SIP Proxy Server 171
18.1.14 Alternative Routing on Detection of Failed SIP Response 171

18.2 SBC Configuration....172
18.2.1 Configuring General Settings....172
18.2.2 Configuring Admission Control....173
18.2.3 Configuring Allowed Coder Groups....175
18.2.4 Configuring SIP Message Policy Rules....176
18.2.5 Routing SBC....178
18.2.5.1 Configuring Classification Rules....178
18.2.5.2 Configuring Condition Rules....182
18.2.5.3 Configuring SBC IP-to-IP Routing....183
18.2.5.4 Configuring Alternative Routing Reasons....189
18.2.6 Dialing Plan Notation for Routing and Manipulation....190
18.2.7 Manipulations SBC....192
18.2.7.1 Configuring Message Manipulations....192
18.2.7.2 Configuring IP-to-IP Inbound Manipulations....195
18.2.7.3 Configuring IP-to-IP Outbound Manipulations....198

19 Stand-Alone Survivability (SAS) Application....201

19.1 Overview....201
19.1.1 SAS Operating Modes....201
19.1.1.1 SAS Outbound Mode....202
19.1.1.2 SAS Redundant Mode....204
19.1.2 SAS Routing....206
19.1.2.1 SAS Routing in Normal State....206
19.1.2.2 SAS Routing in Emergency State....208
19.2 SAS Configuration....209
19.2.1 General SAS Configuration....209
19.2.1.1 Enabling the SAS Application....209
19.2.1.2 Configuring Common SAS Parameters....210
19.2.2 Configuring SAS Outbound Mode....213
19.2.3 Configuring SAS Redundant Mode....214
19.2.4 Advanced SAS Configuration....215
19.2.4.1 Manipulating URI user part of Incoming REGISTER....215
19.2.4.2 Manipulating Destination Number of Incoming INVITE....217
19.2.4.3 SAS Routing Based on IP2IP Routing Table....221
19.2.4.4 Blocking Calls from Unregistered SAS Users....225
19.2.4.5 Configuring SAS Emergency Calls....225
19.2.4.6 Adding SIP Record-Route Header to SIP INVITE....226
19.2.4.7 Replacing Contact Header for SIP Messages....226
19.3 Viewing Registered SAS Users....227
19.4 SAS Cascading....227

20 Overview....231

20.1 Revertive Mode....231
20.2 Initialization Process....232
20.3 HA Status in the Home Page....232
20.4 Device Failure Detection....232

21 HA Configuration....233

21.1 Initial HA Configuration....233
21.1.1 Stage 1: Installation of Both Devices....233
21.1.2 Stage 2: Connect the Devices to Same Network Topology....233
21.1.3 Stage 3: Configure the First Device....233
21.1.4 Stage 4: Configure the Second Device....234
21.1.5 Stage 5: Booting Up Second Device....235
21.2 Configuration while HA State is Operational....235
21.3 Configuring Firewall Allowed Rules....237

22 Troubleshooting HA 238

23 HA Maintenance 239

23.1 Maintenance of the Redundant Device 239
23.2 Replacing a Failed Device 239
23.3 Forcing a Switchover 239
23.4 Software Upgrade 239

24 Basic Maintenance 241

24.1 Resetting the Device 242
24.2 Locking and Unlocking the Device 244
24.3 Saving Configuration 245

25 High Availability Maintenance 246

26 Software Upgrade 247

26.1 Loading Auxiliary Files 247
26.1.1 Auxiliary Configuration Files 249
26.1.1.1 User Information Filefor SBC Users Database 249
26.2 Loading Software Upgrade Key 250
26.3 Software Upgrade Wizard 252
26.4 Backing Up and Loading Configuration File 255

27 System Snapshot 256

27.1 Taking a Snapshot 256
27.2 Returning to a Snapshot State 257

28 Restoring Factory Default Settings 259

28.1 Restoring Defaults using CLI 259
28.2 Restoring Defaults using an ini File 260

29 System Status 262

29.1 Viewing Device Information 262
29.2 Viewing Ethernet Port Information 263

30 Carrier-Grade Alarms 264

30.1 Viewing Active Alarms 264
30.2 Viewing Alarm History 265

31 Performance Monitoring 266

31.1 Viewing Quality of Experience 266
31.2 Viewing Average Call Duration 268

32 VoIP Status 269

32.1 Viewing Active IP Interfaces 269
32.2 Viewing Performance Statistics 270
32.3 Viewing SAS/SBC Registered Users 270
32.4 Viewing Call Routing Status 271

33 Reporting Information to External Party 272

33.1 Generating Call Detail Records 272
33.1.1 CDR Fields for SBC Signaling 272

33.1.2 CDR Fields for SBC Media 273
33.1.3 Supported RADIUS Attributes 274

34 Configuring Syslog Settings 278

Notice

This document describes installation of Alcatel-Lucent OpenTouch™ Session Border Controller.

Information contained in this document is believed to be accurate and reliable at the time of printing. However, due to ongoing product improvements and revisions, Alcatel-Lucent cannot guarantee accuracy of printed material after the Date Published nor can it accept responsibility for errors or omissions. Before consulting this document, check the corresponding Release Notes regarding feature preconditions and/or specific support in this release. In cases where there are discrepancies between this document and the Release Notes, the information in the Release Notes supersedes that in this document. Updates to this document and other documents as well as software files can be downloaded by registered customers at http://www.alcatel-lucent.com.

© Copyright 2012 Alcatel-Lucent. All rights reserved.

This document is subject to change without notice.

Date Published: March-21-2012

Trademarks

Alcatel, Lucent, Alcatel-Lucent and the Alcatel-Lucent logo are trademarks of Alcatel-Lucent. All other trademarks are the property of their respective owners. The information presented is subject to change without notice. Alcatel-Lucent assumes no responsibility for inaccuracies contained herein. Copyright 2012 Alcatel-Lucent. All rights reserved

WEEE EU Directive

Pursuant to the WEEE EU Directive, electronic and electrical waste must not be disposed of with unsorted waste. Please contact your local recycling authority for disposal of this product.

Abbreviations and Terminology

Each abbreviation, unless widely used, is spelled out in full when first used.

Related Documentation

Manual Name
Alcatel-Lucent OpenTouch™ Session Border Controller Release Notes
Alcatel-Lucent OpenTouch™ Session Border Controller Installation Manual
Product Reference Manual for SIP CPE Devices

Note: The scope of this document does not fully cover security aspects for deploying the device in your environment. Security measures should be done in accordance with your organization's security policies.

Note: Throughout this manual, unless otherwise specified, the term *device* refers to the Alcatel-Lucent OpenTouch™ Session Border Controller.

Note: Before configuring the device, ensure that it is installed correctly as instructed in the *Installation Manual*.

1 Introduction

The new Alcatel-Lucent OpenTouch™ Session Border Controller is a pure-software server-based product enabling connectivity and security between enterprises' and Service Providers' VoIP networks.

Alcatel-Lucent OpenTouch™ Session Border Controller provides perimeter defense as a way of protecting companies from malicious VoIP attacks; voice and signaling mediation and normalization for allowing the connection of any PBX and/or IP-PBX to any Service Provider; and service assurance for service quality and manageability.

The device also offers call "survivability" using its Stand Alone Survivability (SAS) application, which ensures service continuity to enterprises served by a centralized SIP-based IP-Centrex server or branch offices of distributed enterprises. SAS enables internal office communication between SIP clients in the case of disconnection from the centralized SIP IP-Centrex server or IP-PBX.

The device allows full management through its HTTP/S-based Web server. This user-friendly Web interface allows remote configuration using any standard Web browser (such as Microsoft™ Internet Explorer™).

Part I

Getting Started

Note: If you are implementing the High Availability feature, see also Part V, "High Availability System" for initial setup.

2 Installing the Software

The Alcatel-Lucent OpenTouch™ Session Border Controller package consists of an Installation CD containing Alcatel-Lucent OpenTouch™ Session Border Controller software, utilities, and related documentation. For installing the Alcatel-Lucent OpenTouch™ Session Border Controller on a server, refer to the *Installation Manual*.

3 Changing Default IP Address to Suit your Network Addressing Scheme

Once you have installed the OpenTouch™ Session Border Controller, you can change its default IP address to suit your network-addressing scheme. Once done, you can connect to the OpenTouch™ Session Border Controller's Web-based management tool (*Web interface*), using this modified IP address.

The default IP addressing schemeis listedbelow:

IP Address: 192.168.0.2

Subnet Mask: 255.255.255.0

Default Gateway IP Address: 192.168.0.1

The procedure below describes how to modify the default IP address, using CLI.

➢ **To reconfigure the IP address, using CLI:**

1. Use the VGA monitor and keyboard to connect to the OpenTouch™ Session Border Controller's CLI management interface.
2. At the prompt, type the username (default is **Admin** - case sensitive), and then press Enter:

   Username: **Admin**
3. At the prompt, type the password (default is **Admin** - case sensitive), and then press Enter:

   Password: **Admin**
4. At the prompt, type the following, and press Enter:

   # enable
5. At the prompt, type the password again and press ENTER:

   Password: **Admin**

6. At the prompt, type the following commands to access the network interface configuration:

```
# configurevoip
(config-voip)# interface network-if 0
(network-if-0)#
```

Note: Use the Tab key to auto-complete partially entered commands.

7. At the prompt, type the following commands to configure the IP address, prefix length and default gateway:

```
(network-if-0)# set ip 10.4.212.155
(network-if-0)# set prefix-length 16
(network-if-0)# set gateway 10.4.0.1
```

8. If OpenTouch™ Session Border Controller is connected to the IP network that uses VLAN ID, type the following command to configure it:

```
(network-if-0)# set vlan-id 10
```

9. At the prompt, type **exit** twice to complete the configuration:

```
(network-if-0)# exit
(config-voip)# exit
```

10. At the prompt, type the following to reset the device and activate the new configuration:

```
# reload
```

4 Configuring Advanced Network Settings using the Web Interface

Once you have assigned an IP address that suits your network environment, you can connect remotely with this IP address to OpenTouch™ Session Border Controller's Web interface for management and configuration.To access the Web interface, see Section 5 on page 17.

For initial setup, it is recommended to configure the following network settings:

To modify and configure IP network interfaces, see Section 11.2on page69

To configure the used physical Ethernet ports (Native VLAN, speed, and mode), see Section 11.1on page68

Part II

Management Tools

This part provides an overview of the various management tools that can be used to configure the device and describes how to configure the management settings. The following management tools can be used to configure the device:

Embedded HTTP/S-based Web server - see 'Web-based Management' on page 17

Command Line Interface (CLI) - see 'CLI-Based Management' on page 46

Configuration *INI* file - see 'INI File-Based Management' on page 53

Simple Network Management Protocol (SNMP) browser software - see 'SNMP-Based Management' on page 47

Note: Some configuration settings can only be done using specific management tools. For example, the *ini* file method provides many parameters that are not supported in the Web interface.

5 Web-Based Management

The device's embedded Web server (hereafter referred to as the *Web interface*) provides FCAPS (fault management, configuration, accounting, performance, and security) functionality. The Web interface allows you to remotely configure the device for quick-and-easy deployment, including the loading of software (.cmp), configuration (.ini), and auxiliary files. The Web interface provides real-time, online monitoring of the device, including display of alarms and their severity. In addition, the Web interface displays performance statistics of voice calls and various traffic parameters.

The Web interface provides a user-friendly, graphical user interface (GUI), which can be accessed using any standard Web browser (e.g., Microsoft™ Internet Explorer). Access to the Web interface is controlled by various security mechanisms such as login user name and password, read-write privileges, and limiting access to specific IP addresses.

Notes:

For a detailed description of all the parameters in the Web interface, see 'Configuration Parameters Reference' on page **Erreur ! Signet non défini.**.

The parameters in the Web interface can alternatively be configured using their corresponding *ini* file parameters, which are enclosed in square brackets "[...]" in 'Configuration Parameters Reference' on page **Erreur ! Signet non défini.**.

The Web interface allows you to configure most of the device's settings. However, additional configuration parameters may exist that are not provided in the Web interface and which can only be configured using *ini* file parameters. These parameters are listed without a corresponding Web parameter name in 'Configuration Parameters Reference' on page **Erreur ! Signet non défini.**.

Some Web interface pages are Software Upgrade Key dependant. These pages appear only if the installed Software Upgrade Key supports the features related to the pages. For viewing your Software Upgrade Key, see 'Loading Software Upgrade Key' on page 250.

5.1 Getting Acquainted with the Web Interface

This section provides a description of the Web interface, including the areas of the GUI, navigation, and configuration methods.

5.1.1 Computer Requirements

The client computer requires the following to work with the Web interface of the device:

A network connection to the device.

One of the following Web browsers:

- Microsoft™ Internet Explorer™ (Version 7.0)
- Mozilla Firefox® (Version8.0)

The following recommended screen resolutions: 1024 x 768 pixels, or 1280 x 1024 pixels.

Note: Your Web browser must be JavaScript-enabled to access the Web interface.

5.1.2 Accessing the Web Interface

The procedure below describes how to access the Web interface.

When initially accessing the Web interface, use

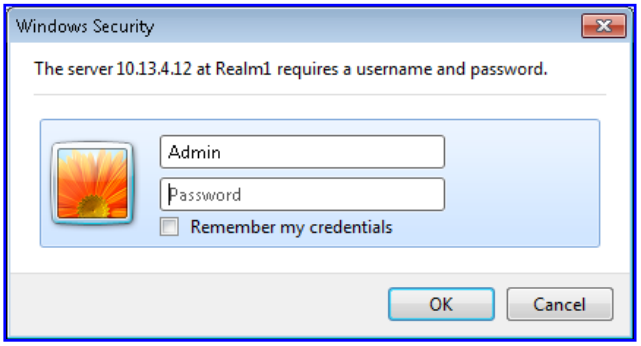
Note: For assigning an IP address to the device, refer to the *Installation Manual*.

To access the Web interface:

1. Open a standard Web browser (see 'Computer Requirements' on page 17).

In the Web browser, specify the IP address of the device (e.g., http://10.1.10.10); the Web interface's Login window appears, as shown below**:**

Figure 5-1: Login Screen



In the 'User Name' and 'Password' fields, enter the case-sensitive, user name and password respectively.

Notes:

The default user name and password is "Admin". To change the login user name and password, see 'Configuring the Web User Accounts' on page 38.

If you want the Web browser to remember your password, select the 'Remember my credentials' check box. The next time you log in to the Web interface, instead of entering your credentials as described in Step 3 above, all you need to do is to click **OK** twice in succession.

Click **OK**; the Web interface is accessed, displaying the Home page (for a detailed description of the Home page, see 'Using the Home Page' on page 36).

Note: If access to the Web interface is denied ("Unauthorized") due to Microsoft Internet Explorer security settings, do the following:

Delete all cookies in the Temporary Internet Files folder. If this does not resolve the problem, the security settings may need to be altered (continue with Step 2).

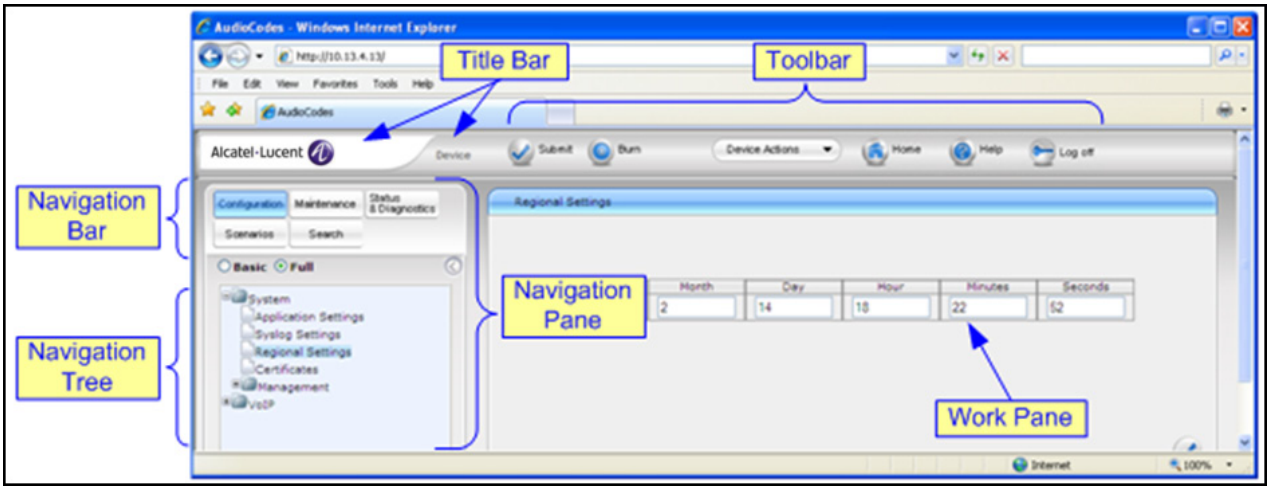
In Internet Explorer, navigate to **Tools** menu >**Internet Options**>**Security** tab >**Custom Level**, and then scroll down to the Logon options and select **Prompt for username and password**. Select the **Advanced** tab, and then scroll down until the HTTP 1.1 Settings are displayed and verify that **Use HTTP 1.1** is selected.

Quit the Web browser and start it again.

5.1.3 Areas of the GUI

The figure below displays the areas of the Web interface GUI:

Figure 5-2: Areas of the Web GUI



The Web GUI consists of the following main areas:

Title bar: Displays the corporate logo image and product name.

Toolbar: Provides frequently required command buttons (see 'Toolbar Description' on page 20).

Navigation Pane: Includes the following areas:

Navigation bar: Provides tabs for accessing the configuration menus (see 'Navigation Tree' on page 20) and searching Web interface parameters (see 'Searching for Configuration Parameters' on page 31).

Navigation tree: Displays the elements pertaining to the selected tab on the Navigation bar (tree-like structure of the configuration menus or Search engine).

Work pane: Displays configuration pages in which configuration is done (see 'Working with Configuration Pages' on page 24).

5.1.4 Toolbar Description

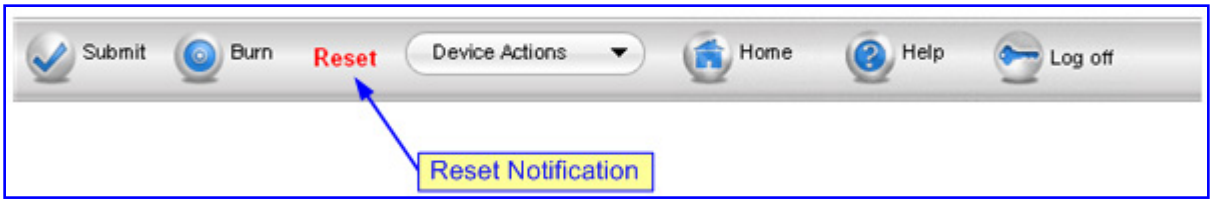
The toolbar provides frequently required command buttons, as described in the table below:

Table 5-1: Description of Toolbar Buttons

Icon	Button Name	Description
	Submit	Applies parameter settings to the device (see 'Saving Configuration' on page 245). **Note:** This icon is grayed out when not applicable to the currently opened page.
	Burn	Saves parameter settings to flash memory (see 'Saving Configuration' on page 245).
	Device Actions	Opens a drop-down menu list with frequently needed commands: **Load Configuration File:** opens the Configuration File page for loading an *ini* file (see 'Backing Up and Loading Configuration File' on page 255). **Save Configuration File:** opens the Configuration File page for saving the *ini* file to a folder on a computer (see 'Backing Up and Loading Configuration File' on page 255). **Reset:** opens the Maintenance Actions page for resetting the device (see 'Resetting the Device' on page 242). **Software Upgrade Wizard:** starts the Software Upgrade wizard for upgrading the device's software (see 'Software Upgrade Wizard' on page 252).
	Home	Opens the Home page (see 'Using the Home Page' on page 36).
	Help	Opens the Online Help topic of the currently opened configuration page (see 'Getting Help' on page 34).
	Log off	Logs off a session with the Web interface (see 'Logging Off the Web Interface' on page 35).

Note: If you modify parameters that take effect only after a device reset, after you click the **Submit** button, the toolbar displays "Reset" (in red color), as shown in the figure below. This is a reminder that you need to later save your settings to flash memory and reset the device.

Figure 5-3: "Reset" Displayed on Toolbar



5.1.5 Navigation Tree

The Navigation tree is located in the Navigation pane. It displays the menus pertaining to the selected menu tab on the Navigation bar and is used for accessing the configuration pages.

The Navigation tree displays a tree-like structure of menus. You can drill-down to the required page item level to open its corresponding page in the Work pane.

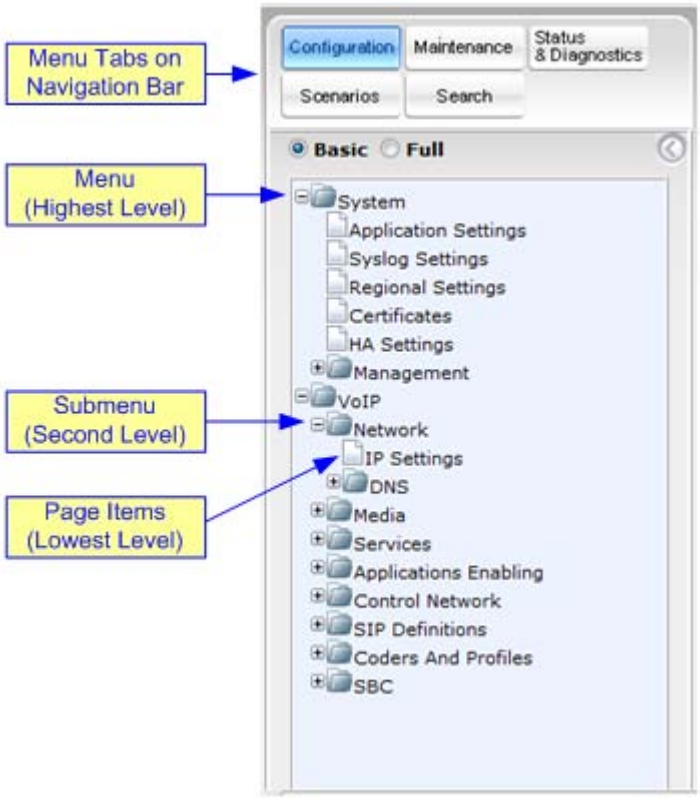
The terminology used throughout this manual for referring to the hierarchical structure of the tree is as follows:

menu: first level (highest level)

submenu: second level - contained within a menu

page item: last level (lowest level in a menu) - contained within a menu or submenu

Figure 5-4: Navigation Tree



To view menus in the Navigation tree:

On the Navigation bar, select the required tab - **Configuration**, **Maintenance**, or **Status & Diagnostics**.

To navigate to a page:

1. Navigate to the required page item, by performing the following:

   Drilling-down using the **plus** ⊞ sign to expand the menu and submenus.

   Drilling-up using the **minus** ⊟ sign to collapse the menu and submenus.

Select the required page item; the page opens in the Work pane.

5.1.5.1 Displaying Navigation Tree in Basic and Full View

You can view an expanded or reduced Navigation tree display regarding the number of listed menus and submenus. This is relevant when using the configuration tabs (**Configuration**, **Maintenance**, and **Status & Diagnostics**) on the Navigation bar.

The Navigation tree menu can be displayed in one of two views:

Basic: displays only commonly used menus

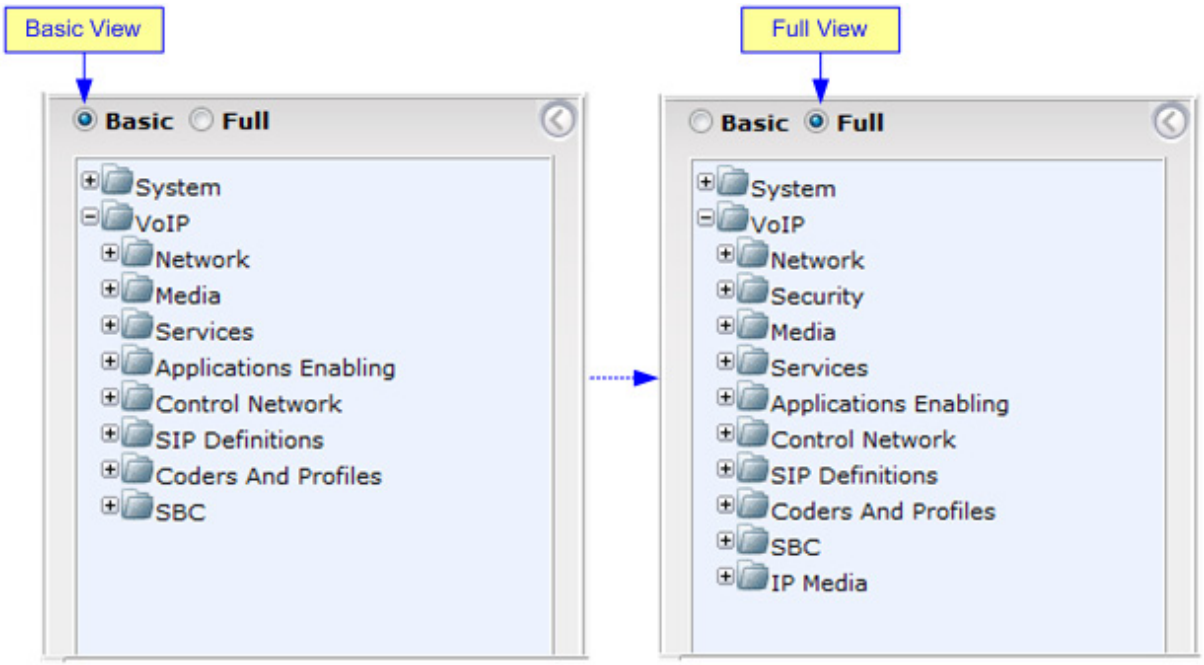
Full: displays all the menus pertaining to a configuration tab

The advantage of the Basic view is that it prevents "cluttering" of the Navigation tree with menus that may not be required. Therefore, a Basic view allows you to easily locate required menus.

To toggle between Full and Basic view:

Select the **Basic** option, located below the Navigation bar, to display a reduced menu tree; select the **Full** option to display all the menus. By default, the **Basic** option is selected.

Figure 5-5: Basic and Full View



Note: After you reset the device, the Web GUI is displayed in Basic view.

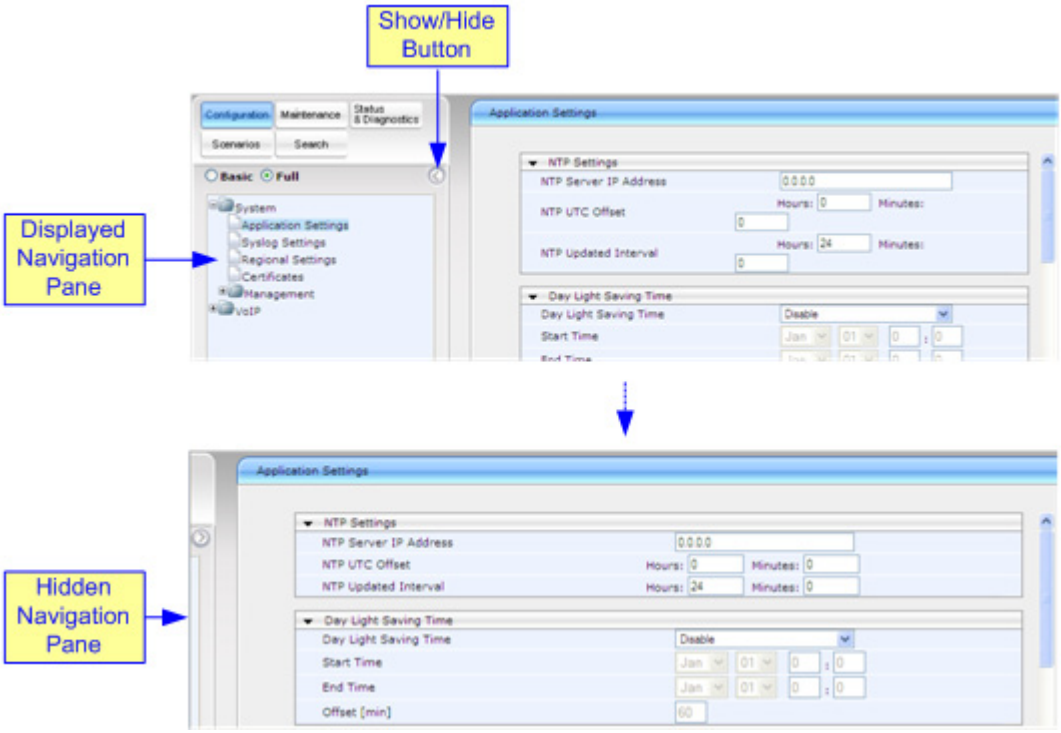
5.1.5.2 Showing / Hiding the Navigation Pane

The Navigation pane can be hidden to provide more space for elements displayed in the Work pane. This is especially useful when the Work pane displays a table that's wider than the Work pane and to view all the columns, you need to use scroll bars. The arrow button located just below the Navigation bar is used to hide and show the Navigation pane.

To hide the Navigation pane: click the left-pointing arrow ; the pane is hidden and the button is replaced by the right-pointing arrow button.

To show the Navigation pane: click the right-pointing arrow ; the pane is displayed and the button is replaced by the left-pointing arrow button.

Figure 5-6: Showing and Hiding Navigation Pane



5.1.6 Working with Configuration Pages

The configuration pages contain the parameters for configuring the device and are displayed in the Work pane, located to the right of the Navigation pane.

5.1.6.1 Accessing Pages

The configuration pages are accessed by clicking the required page item in the Navigation tree.

To open a configuration page:

1. On the Navigation bar, click the required tab:

   Configuration

   Maintenance

   Status & Diagnostics

   The menus pertaining to the selected tab appear in the Navigation tree.

In the Navigation tree, drill-down to the required submenu and then click the required page item; the page opens in the Work pane.

You can also access previously opened pages by clicking the Web browser's **Back** button until you have reached the required page. This is useful if you want to view pages in which you have performed configurations in the current Web session.

Notes:

- You can also access certain pages from the **Device Actions** button located on the toolbar (see 'Toolbar Description' on page 20).
- To view all the menus in the Navigation tree, ensure that the Navigation tree is in Full view (see 'Displaying Navigation Tree in Basic and Full View' on page 21).
- To get Online Help for the currently displayed page, see 'Getting Help' on page 34.
- Certain pages may not be accessible or may be read-only if your Web user account's access level is low (see 'Configuring the Web User Accounts' on page 38). If a page is read-only, 'Read-Only Mode' is displayed at the bottom of the page.

5.1.6.2 Viewing Parameters

For convenience, some pages allow you to view a reduced or expanded display of parameters. The Web interface provides two methods for displaying page parameters:

Displaying "basic" and "advanced" parameters - see 'Displaying Basic and Advanced Parameters' on page 25

Displaying parameter groups - see 'Showing / Hiding Parameter Groups' on page 26

5.1.6.2.1 Displaying Basic and Advanced Parameters

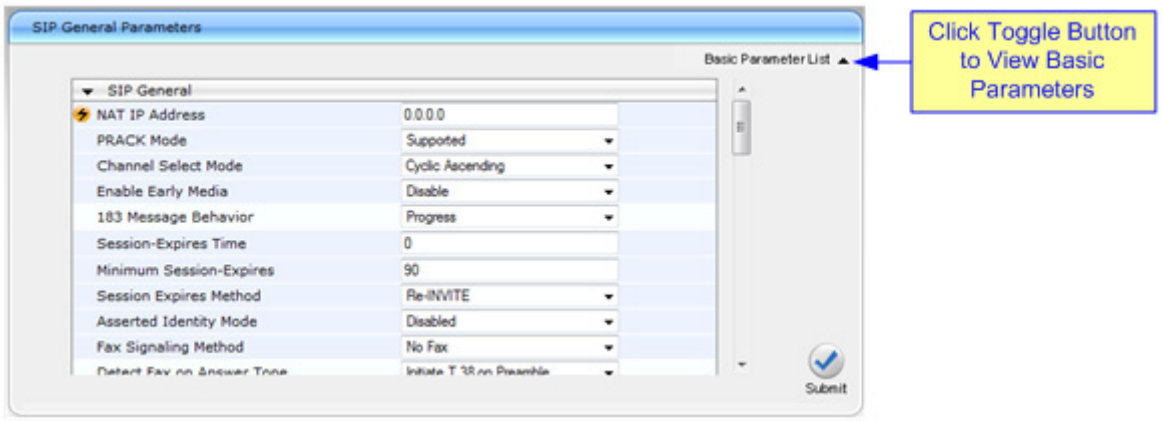
Some pages provide you with an **Advanced Parameter List** / **Basic Parameter List** toggle button that allows you to show or hide advanced parameters (in addition to displaying the basic parameters). This button is located on the top-right corner of the page and has two states:

Advanced Parameter List button with down-pointing arrow: click this button to display all parameters.

Basic Parameter List button with up-pointing arrow: click this button to show only common (*basic*) parameters.

The figure below shows an example of a page displaying basic parameters only, and then showing advanced parameters as well, using the **Advanced Parameter List** button.

Figure 5-7: Toggling between Basic and Advanced View



For ease of identification, the basic parameters are displayed with a darker blue color background than the advanced parameters.

Notes:

When the Navigation tree is in Full mode (see 'Navigation Tree' on page 20), configuration pages display all their parameters (i.e., the Advanced Parameter List view is displayed).

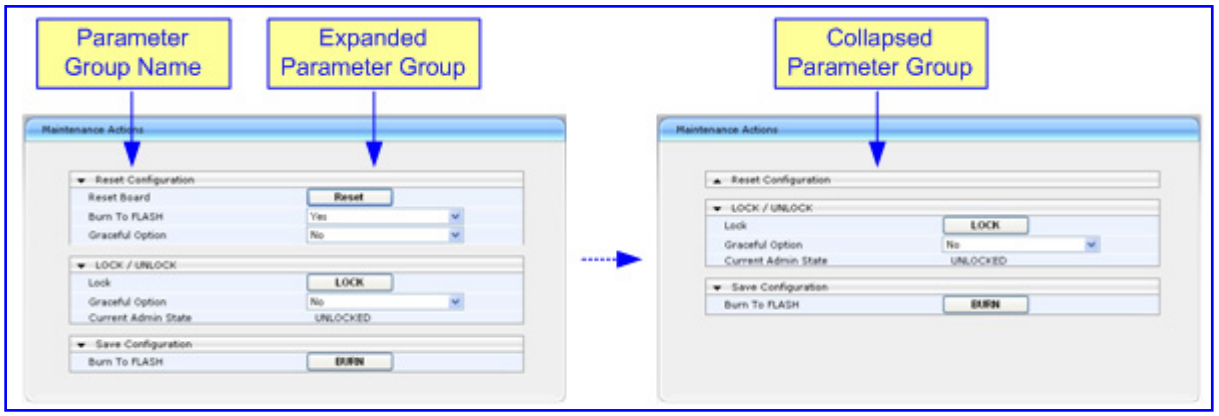
If a page contains only basic parameters, the **Basic Parameter List** button is not displayed.

After you reset the device, the Web pages display only the basic parameters.

5.1.6.2.2 Showing / Hiding Parameter Groups

Some pages provide groups of parameters, which can be hidden or shown. To toggle between hiding and showing a group, simply click the group title button that appears above each group. The button appears with a down-pointing or up-pointing arrow, indicating that it can be collapsed or expanded when clicked, respectively.

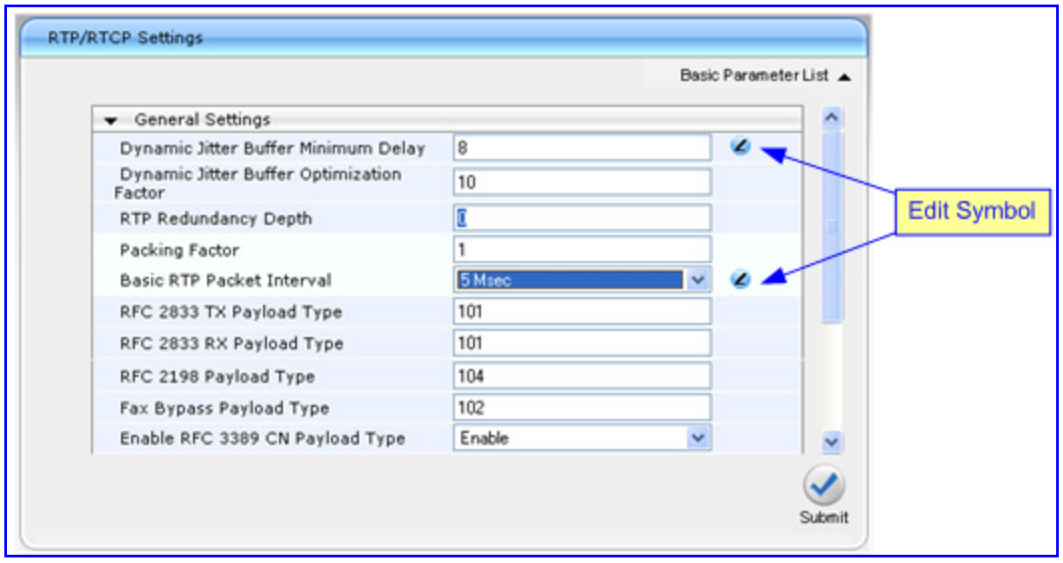
Figure 5-8: Expanding and Collapsing Parameter Groups



5.1.6.3 Modifying and Saving Parameters

When you modify a parameter value on a page, the **Edit** symbol appears to the right of the parameter. This is useful for indicating the parameters that you have currently modified (before applying the changes). After you apply your modifications, the symbols disappear.

Figure 5-9: Edit Symbol after Modifying Parameter Value



To save configuration changes on a page to the device's volatile memory (RAM), do one of the following:

On the toolbar, click the **Submit** button.

At the bottom of the page, click the **Submit** button.

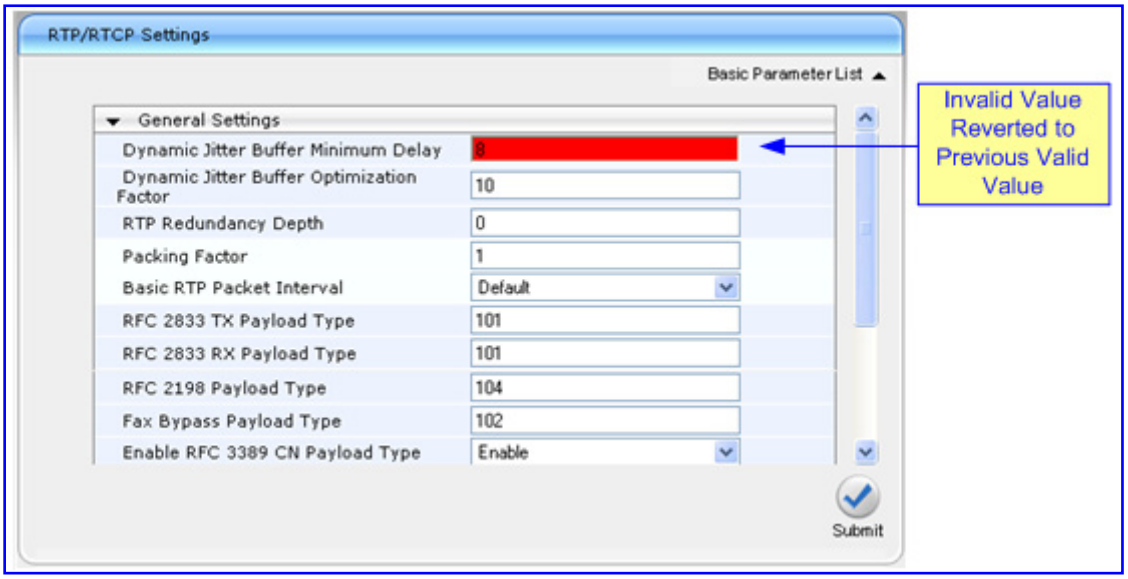
When you click **Submit**, modifications to parameters with on-the-fly capabilities are immediately applied to the device and take effect; other parameters displayed on the page with the lightning symbol are not changeable on-the-fly and require a device reset (see 'Resetting the Device' on page 242) before taking effect.

Notes:

- Parameters saved to the volatile memory (by clicking **Submit**), revert to their previous settings after a hardware or software reset (or if the device is powered down). Therefore, to ensure parameter changes (whether on-the-fly or not) are retained, save ('burn') them to the device's non-volatile memory, i.e., flash (see 'Saving Configuration' on page 245).
- If you modify a parameter value and then attempt to navigate away from the page without clicking **Submit**, a message box appears notifying you of this. Click **Yes** to save your modifications or **No** to ignore them.

If you enter an invalid parameter value (e.g., not in the range of permitted values) and then click **Submit**, a message box appears notifying you of the invalid value. In addition, the parameter value reverts to its previous value and is highlighted in red, as shown in the figure below:

Figure 5-10: Value Reverts to Previous Valid Value



5.1.6.4 Entering Phone Numbers

Phone numbers or prefixes that you need to configure throughout the Web interface must be entered only as digits without any other characters. For example, if you wish to enter the phone number 555-1212, it must be entered as 5551212 without the hyphen (-). If the hyphen is entered, the entry is invalid.

5.1.6.5 Working with Tables

This section describes how to work with configuration tables, which are provided in basic or enhanced design (depending on the configuration page).

5.1.6.5.1 Basic Design Tables

The basic design tables provide the following command buttons:

Add Index: adds an index entry to the table.

Duplicate: duplicates a selected, existing index entry.

Compact: organizes the index entries in ascending, consecutive order.

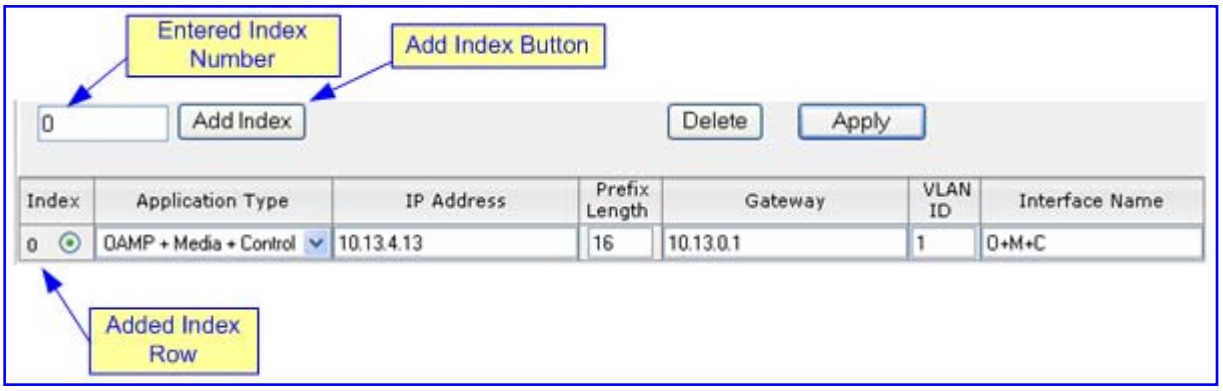
Delete: deletes a selected index entry.

Apply: saves the configuration.

To add an entry to a table:

1. In the 'Add Index' field, enter the desired index entry number, and then click **Add Index**; an index entry row appears in the table:

Figure 5-11: Adding an Index Entry to a Table



Click **Apply** to save the index entry.

Notes:

Before you can add another index entry, ensure that you have applied the previously added index entry (by clicking **Apply**).

If you leave the 'Add' field blank and then click **Add Index**, the existing index entries are all incremented by one and the newly added index entry is assigned the index 0.

To copy an existing index table entry:

1. In the 'Index' column, select the index that you want to duplicate; the **Edit** button appears.

Click **Edit**; the fields in the corresponding index row become available.

Click **Duplicate**; a new index entry is added with identical settings as the selected index in Step 1. In addition, all existing index entries are incremented by one and the newly added index entry is assigned the index 0.

To edit an index table entry:

1. In the 'Index' column, select the index corresponding to the table row that you want to edit.

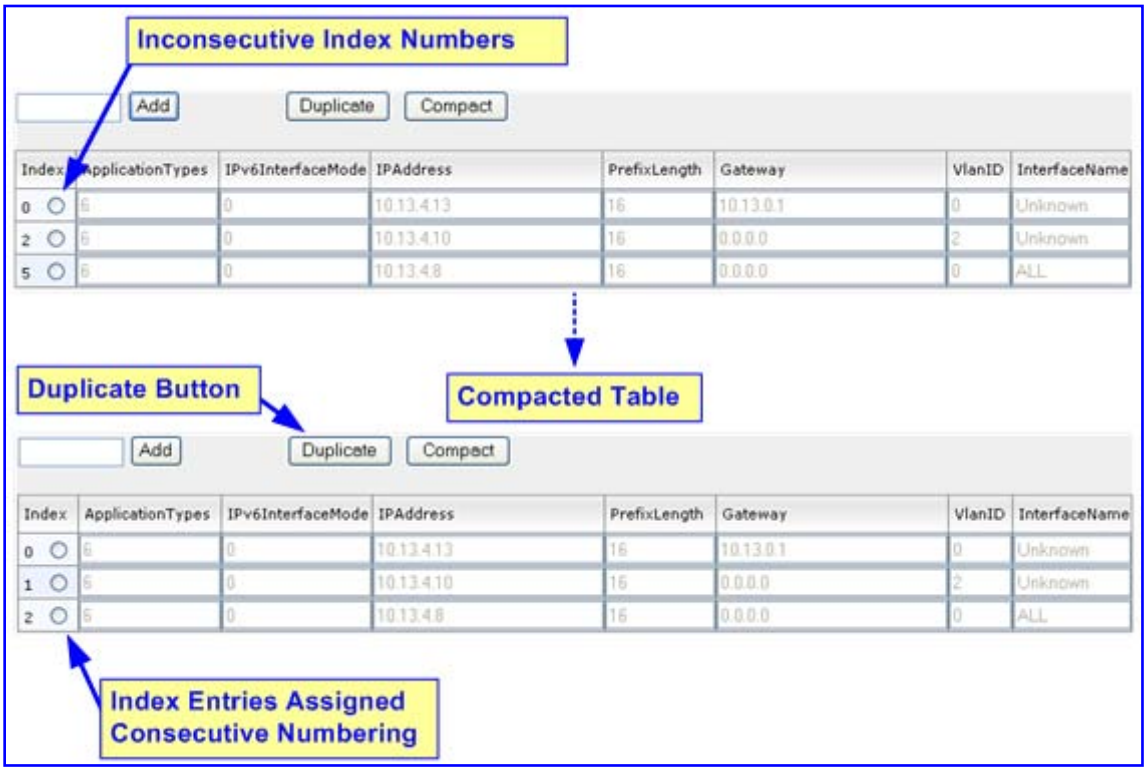
Click **Edit**; the fields in the corresponding index row become available.

Modify the values as required, and then click **Apply**; the new settings are applied.

To organize the index entries in ascending, consecutive order:

Click **Compact**; the index entries are organized in ascending, consecutive order, starting from index 0. For example, if you added three index entries 0, 4, and 6, then the index entry 4 is re-assigned index number 1 and the index entry 6 is re-assigned index number 2.

Figure 5-12: Compacting a Web Interface Table



To delete an index table entry:

1. In the 'Index' column, select the index corresponding to the table row that you want to delete.

Click **Delete**; the table row is removed from the table.

5.1.6.5.2 Enhanced Design Tables

The enhanced table structure includes the following buttons:

Add: adds a row entry to the table

Edit: edits the selected table row

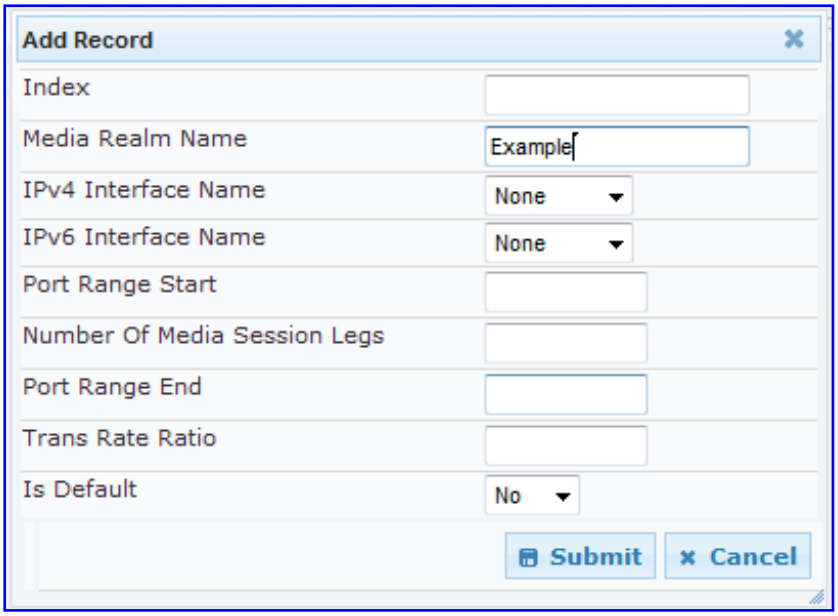
Delete: deletes a selected table row

View/Unview: shows or hides all configuration settings of selected table rows

To add an entry:

1. Click the **Add** button; the Add Record dialog box appears:

Figure 5-13: Add Record Dialog Box



Configure the required parameters, and then click **Submit** to apply your changes (or **Cancel** to ignore your changes); the new row entry is added to the table. If the configuration is invalid, the index of the table row is highlighted in red, as shown below:

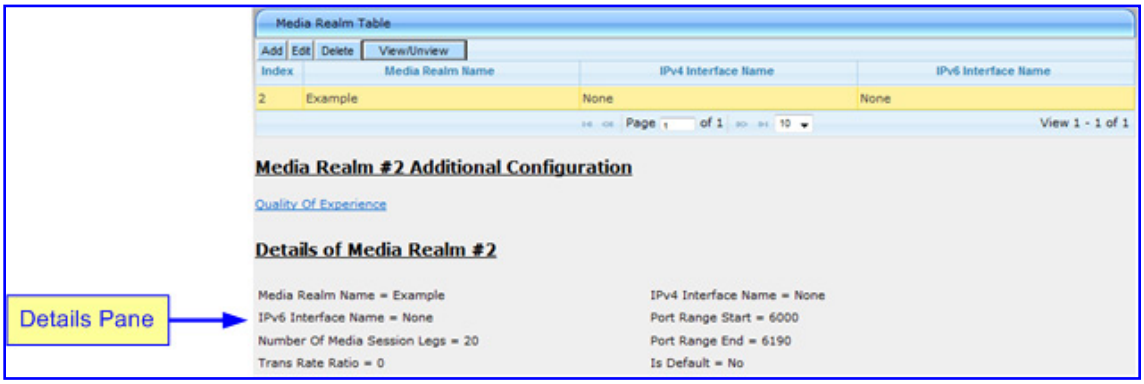
Figure 5-14: Index Highlighted in Red

By default, the table displays 10 entries per page. However, you can change this to 5 by selecting **5** from the drop-down list located immediately below the table. If your table spans over multiple pages, you can navigate between the pages by clicking the left and right arrow buttons located immediately below the table.

To view the configuration settings of an entry:

1. Select the table row that you want to view, and then click the **View/Unview** button; a Details pane appears below the table, displaying the configuration settings of the selected row, as shown below:

Figure 5-15: Displayed Details Pane



To hide the Details pane, click the **View/Unview** button again.

To edit an entry:

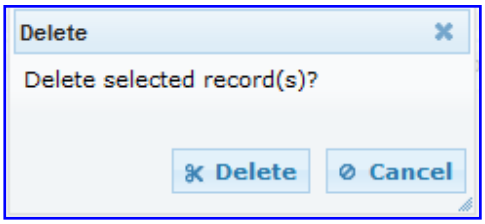
1. Select the table row that you want to modify, and then click the **Edit** button; the Edit Record dialog box appears.

Make the required changes, and then click **Submit**.

To delete an entry:

1. Select the table row that you want to delete, and then click the **Delete** button; the Delete message box appears:

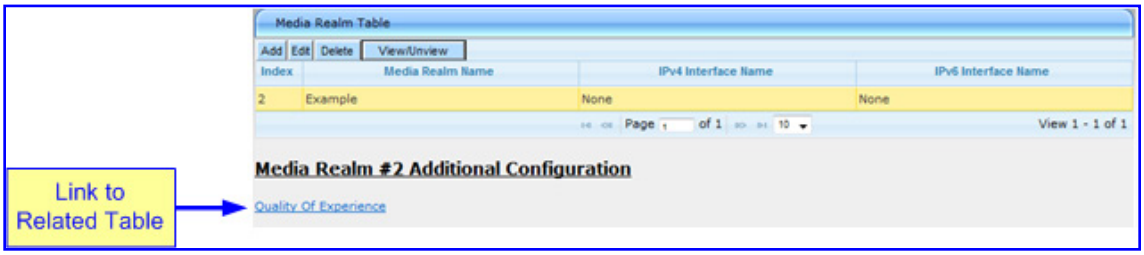
Figure 5-16: Delete Message Box



Click **Delete** to confirm deletion (or **Cancel** to abort the process).

Some tables provide a link to a related table for advanced configuration of a selected row entry, as shown below:

Figure 5-17: Link to Related Table



5.1.7 Searching for Configuration Parameters

The Web interface provides a search engine that allows you to search any *ini* file parameter that is configurable in the Web interface (i.e., has a corresponding Web parameter). You can search for a specific parameter (e.g., "EnableIPSec") or a substring of that parameter (e.g.,

"sec"). If you search for a substring, all parameters containing the searched substring in their names are listed.

To search for *ini* file parameters configurable in the Web interface:

1. On the Navigation bar, click the **Search** tab; the Search engine appears in the Navigation pane.

In the 'Search' field, enter the parameter name or substring of the parameter name that you want to search. If you have done a previous search for such a parameter, instead of entering the required string, you can use the 'Search History' drop-down list to select the string saved from a previous search.

Click **Search**; a list of located parameters based on your search appears in the Navigation pane. Each searched result displays the following:

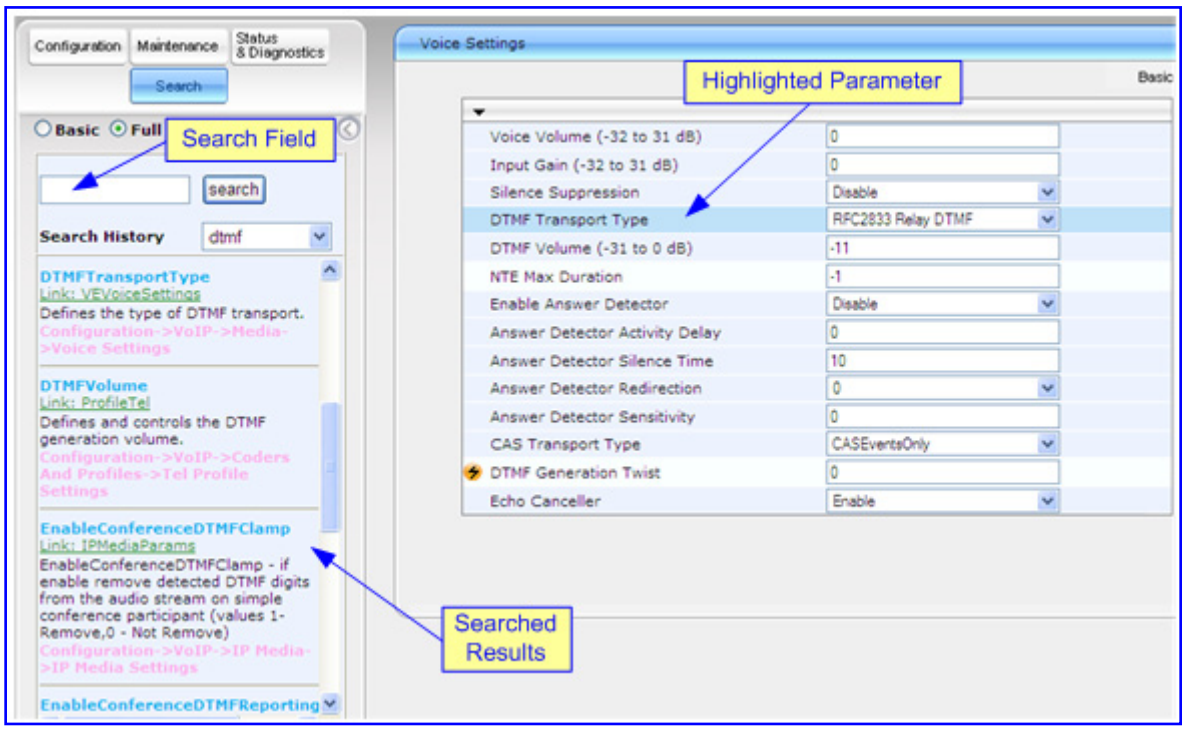
ini file parameter name

Link (in green) to its location (page) in the Web interface

Brief description of the parameter

In the searched list, click the required parameter (link in green) to open the page in which the parameter appears; the relevant page opens in the Work pane and the searched parameter is highlighted in the page for easy identification, as shown in the figure below:

Figure 5-18: Searched Result Screen



5.1.8 Creating a Login Welcome Message

You can create a Welcome message box (alert message) that appears after each successful login to the Web interface. The *WelcomeMessage* ini file parameter table allows you to create the Welcome message. Up to 20 lines of character strings can be defined for the message. If this parameter is not configured, no Welcome message box is displayed after login.

An example of a Welcome message is shown in the figure below:

Figure 5-19: User-Defined Web Welcome Message after Login

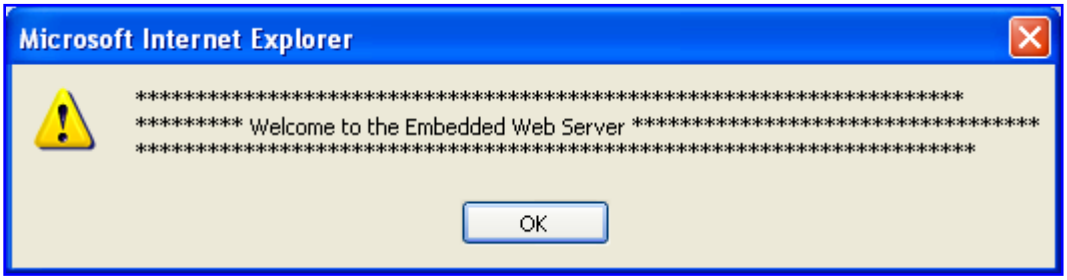


Table 5-2: ini File Parameter for Welcome Login Message

Parameter	Description
WelcomeMessage	Defines the Welcome message that appears after a successful login to the Web interface. The format of this parameter is as follows: [WelcomeMessage] FORMAT WelcomeMessage_Index = WelcomeMessage_Text; [\WelcomeMessage] For Example: [WelcomeMessage] FORMAT WelcomeMessage_Index = WelcomeMessage_Text; WelcomeMessage 1 = "**********"; WelcomeMessage 2 = "********* This is a Welcome message **"; WelcomeMessage 3 = "**********"; [\WelcomeMessage] **Note:** Each index represents a line of text in the Welcome message box. Up to 20 indices can be defined.

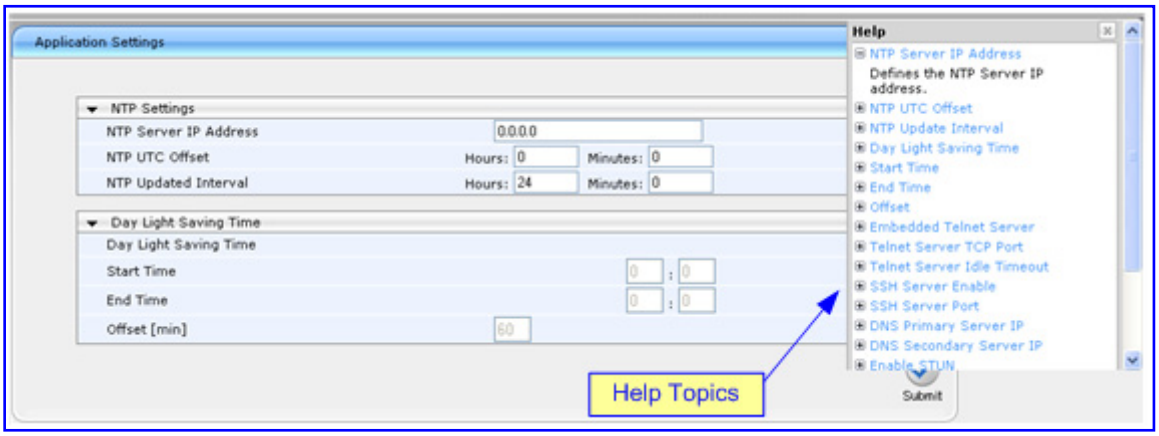
5.1.9 Getting Help

The Web interface provides you with context-sensitive Online Help. The Online Help provides brief descriptions of parameters pertaining to the currently opened page.

To view the Help topic of a currently opened page:

1. On the toolbar, click the **Help** button; the Help topic pertaining to the opened page appears, as shown below:

Figure 5-20: Help Topic for Current Page



To view a description of a parameter, click the **plus** ⊞ sign to expand the parameter. To collapse the description, click the **minus** ⊟ sign.

To close the Help topic, click the **close** button located on the top-right corner of the Help topic window or simply click the **Help** button.

Note: Instead of clicking the **Help** button for each page you open, you can open it once for a page and then simply leave it open. Each time you open a different page, the Help topic pertaining to that page is automatically displayed.

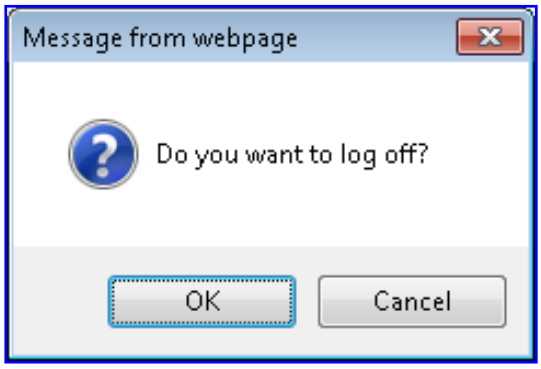
5.1.10 Logging Off the Web Interface

You can log off the Web interface and re-access it with a different user account. For more information on Web User Accounts, see 'Configuring Web User Accounts' on page 38.

To log off the Web interface:

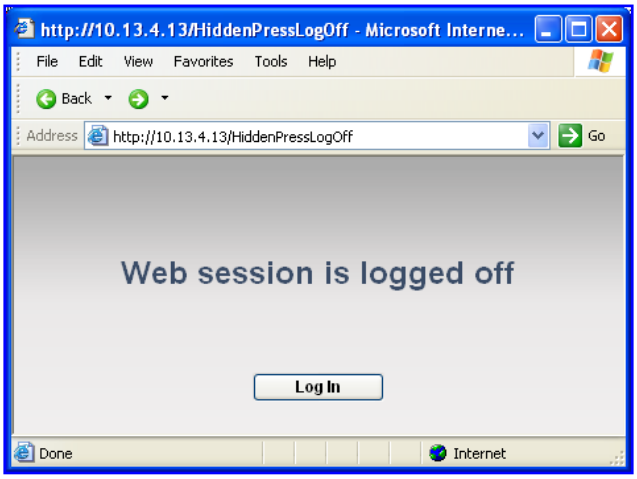
1. On the toolbar, click the **Log Off** button; the Log Off confirmation message box appears:

Figure 5-21: Log Off Confirmation Box



Click **OK**; the Web session is logged off and the **Log In** button appears.

Figure 5-22: Web Session Logged Off



To log in again, simply click the **Log In** button, and then in the Login window, enter your user name and password (see 'Accessing the Web Interface' on page 18).

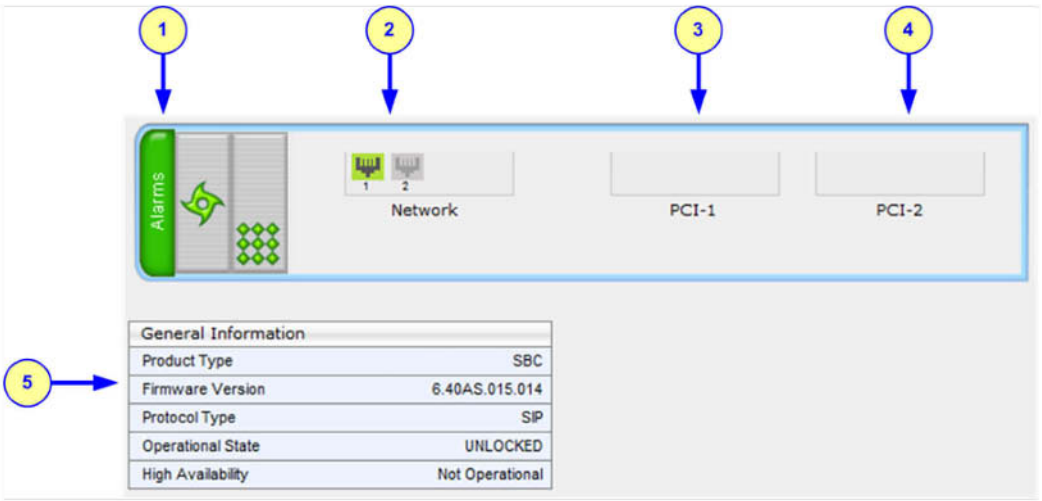
5.2 Using the Home Page

By default, the Home page is displayed when you access the device's Web interface. The Home page provides you with a graphical display of the device's front panel, displaying color-coded status icons for monitoring the functioning of the device. The Home page also displays general device information (in the 'General Information' pane) such as the device's IP address and firmware version.

To access the Home page:

On the toolbar, click the **Home** icon.

Figure 5-23: Home Page



Note: The displayed number of Ethernet ports depends on your hardware configuration.

Table 5-3: Description of the Areas of the Home Page

Item #	Description
1	Displays the highest severity of an active alarm raised (if any) by the device: Green = No alarms Red = Critical alarm Orange = Major alarm Yellow = Minor alarm To view a list of active alarms in the Active Alarms page (see Viewing Active Alarms on page 264), click the Alarms area.
2	Gigabit Ethernet LAN port status icons: (green): Ethernet link is working (gray): Ethernet link is not connected You can view detailed Ethernet port information, by clicking these icons. When clicked, the Ethernet Port Information page is displayed(see Viewing Ethernet Port Information on page 263).

Item #	Description
3 & 4	Reserved for future use.
5	General Information pane displaying the following: **Firmware Version:** software version currently running on the device **Protocol Type:** signaling protocol currently used by the device (i.e. SIP) **Gateway Operational State:** operational state of the device: "LOCKED" - device is locked (i.e. no new calls are accepted) "UNLOCKED" - device is not locked "SHUTTING DOWN" - device is currently shutting down **High Availability:** status of the device's HA mode: "Not Operational": HA is not configured or device not installed with HA Feature Key "Synchronizing": Redundant device synchronizing with Active device "Operational": Device is in HA mode "Stand Alone": HA is configured but Redundant device is missing and HA is currently unavailable "Not Available": HA is not configured correctly (error)

5.3 Configuring Web User Accounts

To prevent unauthorized access to the Web interface, two Web user accounts are available (primary and secondary) with assigned user name, password, and access level. When you login to the Web interface, you are requested to provide the user name and password of one of these Web user accounts. If the Web session is idle (i.e., no actions are performed) for more than five minutes, the Web session expires and you are once again requested to login with your user name and password. Up to five Web users can simultaneously open (log in to) a session on the device's Web interface. Users can be banned for a period of time upon a user-defined number of unsuccessful login attempts. Login information (such as how many login attempts were made and the last successful login time) can be presented to the user.

Each Web user account is composed of three attributes:

User name and password: enables access (login) to the Web interface.

Access level: determines the extent of the access (i.e., availability of pages and read / write privileges). The available access levels and their corresponding privileges are listed in the table below:

Table 5-4: Web User Accounts Access Levels and Privileges

Access Level	Numeric Representation*	Privileges
Security Administrator	200	Read / write privileges for all pages.
Administrator	100	Read / write privileges for all pages except security-related pages, which are read-only.
User Monitor	50	No access to security-related and file-loading pages; read-only access to the other pages. This read-only access level is typically applied to the secondary Web user account.
No Access	0	No access to any page.
* The numeric representation of the access level is used only to define accounts in a RADIUS server (the access level ranges from 1 to 255).		

The default attributes for the two Web user accounts are shown in the following table:

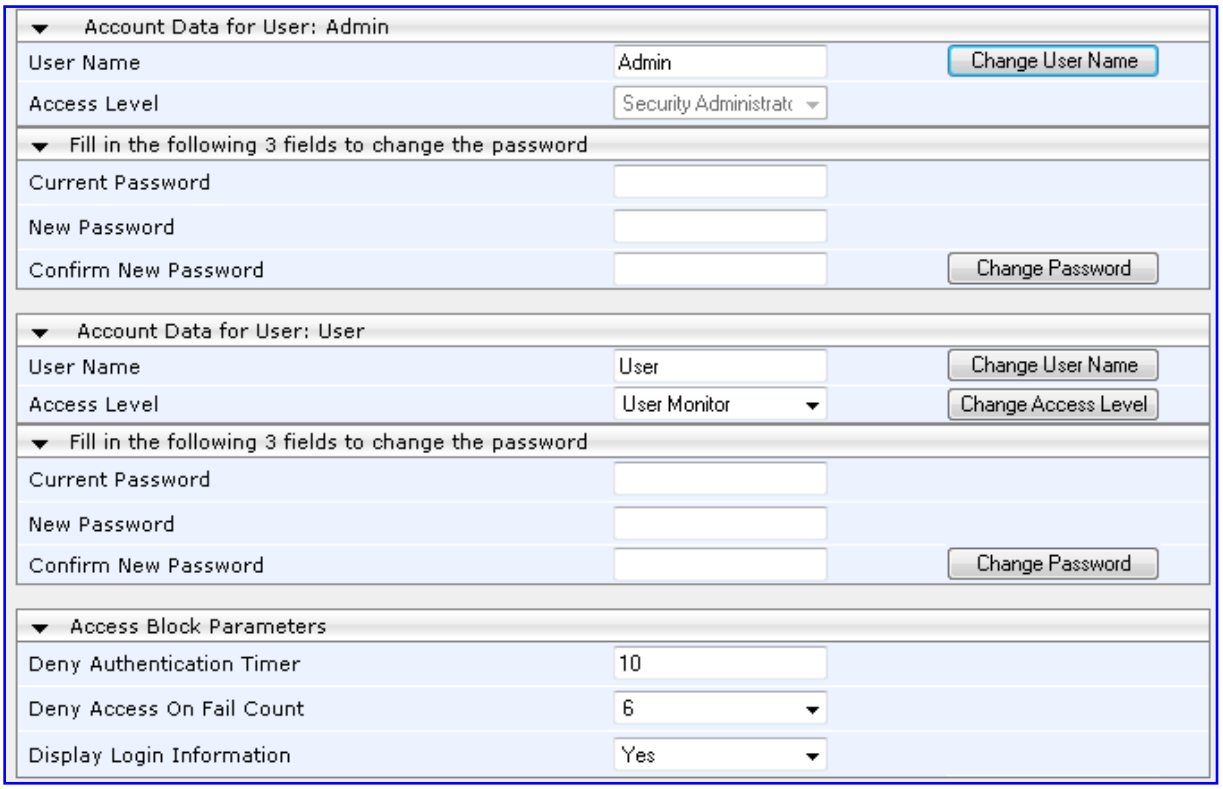
Table 5-5: Default Attributes for the Web User Accounts

Account / Attribute	User Name (Case-Sensitive)	Password (Case-Sensitive)	Access Level
Primary Account	Admin	Admin	Security Administrator **Note:** The Access Level cannot be changed for this account type.
Secondary Account	User	User	User Monitor

To change the Web user accounts attributes:

1. Open the Web User Accounts page (**Configuration** tab >**System** menu >**Web User Accounts**).

Figure 5-24: WEB User Accounts Page (for Users with 'Security Administrator' Privileges)



Note: If you are logged into the Web interface as the Security Administrator, both Web user accounts are displayed on the Web User Accounts page (as shown above). If you are logged in with the secondary user account, only the details of the secondary account are displayed on the page.

To change the access level of the secondary account:

a. From the 'Access Level' drop-down list, select the new access level.

Click **Change Access Level**; the new access level is applied immediately.

Notes:

The access level of the primary Web user account is 'Security Administrator', which cannot be modified.

The access level of the secondary account can only be modified by the primary account user or a secondary account user with 'Security Administrator' access level.

To change the user name of an account, perform the following:

a. In the field 'User Name', enter the new user name (maximum of 19 case-sensitive characters).

Click **Change User Name**; if you are currently logged into the Web interface with this account, the 'Enter Network Password' dialog box appears, requesting you to enter the new user name.

To change the password of an account, perform the following:

a. In the field 'Current Password', enter the current password.

In the fields 'New Password' and 'Confirm New Password', enter the new password (maximum of 19 case-sensitive characters).

Click **Change Password**; if you are currently logged into the Web interface with this account, the 'Enter Network Password' dialog box appears, requesting you to enter the new password.

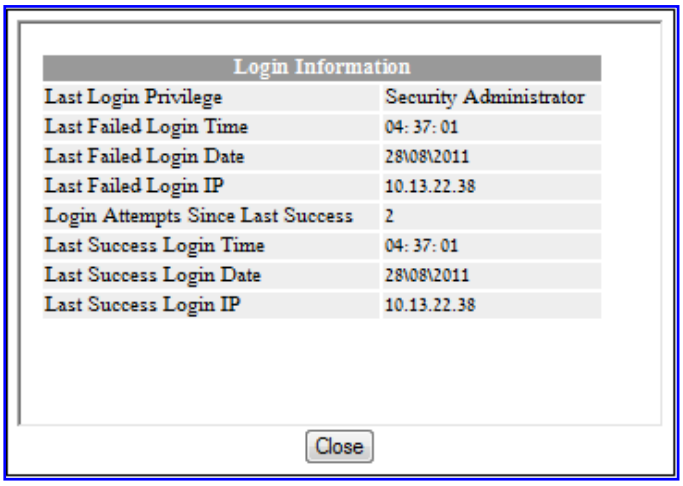
To prevent user access after a specific number of failed logins, do the following:

a. From the 'Deny Access On Fail Count' drop-down list, select the number of failed logins after which the user is prevented access to the device for a user-defined time (see next step).

In the 'Deny Authentication Timer' field, enter the interval (in seconds) that the user needs to wait before a new login attempt from the same IP address can be done after reaching the number of failed login attempts (defined in the previous step).

To display user login information upon a successful login, from the 'Display Login Information' drop-down list, select **Yes**. After you login, the following window is displayed:

Figure 5-25: Login Information Window



Click **Submit** to apply your changes.

Notes:

For security, it's recommended that you change the default user name and password.

A Web user with access level 'Security Administrator' can change all attributes of all the Web user accounts. Web users with an access level other than 'Security Administrator' can only change their own password and user name.

To reset the two Web user accounts' user names and passwords to default, set the *ini* file parameter ResetWebPassword to 1.

To access the Web interface with a different account, click the **Log off** button located on the toolbar, click any button or page item, and then re-access the Web interface with a different user name and password.

You can set the entire Web interface to read-only (regardless of Web user account's access level), by using the *ini* file parameter DisableWebConfig (see 'Web and Telnet Parameters' on page 290).

Access to the Web interface can be disabled, by setting the ini file parameter DisableWebTask to 1. By default, access is enabled.

You can define additional Web user accounts using a RADIUS server (refer to the *Product Reference Manual*).

For secured HTTP connection (HTTPS), refer to the *Product Reference Manual*.

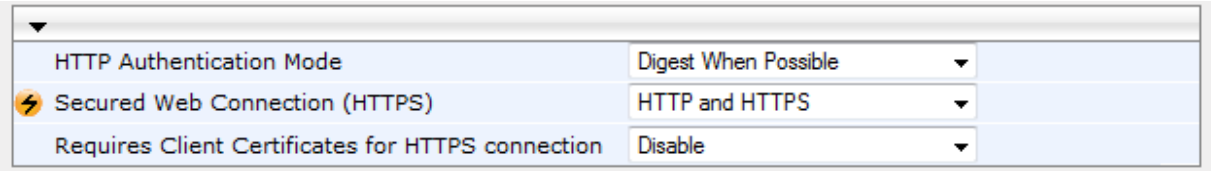
5.4 Configuring Web Security Settings

The WEB Security Settings page is used to define a secure Web access communication method. For a description of these parameters, see 'Web and Telnet Parameters' on page 290.

To define Web access security:

1. Open the WEB Security Settings page (**Configuration** tab >**System** menu >**Management** submenu >**WEB Security Settings**).

Figure 5-26: Web Security Page



Configure the parameters as required.

Click Subm**it to** apply your changes.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

5.5 Web Login Authentication using Smart Cards

You can enable Web login authentication using certificates from a third-party, common access card (CAC) with user identification. When a user attempts to access the device through the Web browser (HTTPS), the device retrieves the Web user's login username (and other information, if required) from the CAC. The user attempting to access the device is only required to provide the login password. Typically, a TLS connection is established between the CAC and the device's Web interface, and a RADIUS server is implemented to authenticate the password with the username. Therefore, this feature implements a two-factor authentication - what the user has (i.e., the physical card) and what the user knows (i.e., the login password).

This feature is enabled using the EnableMgmtTwoFactorAuthentication parameter.

Note: For specific integration requirements for implementing a third-party smart card for Web login authentication, contact your representative.

To login to the Web interface using CAC:

1. Insert the Common Access Card into the card reader.

Access the device using the following URL: https://<host name or IP address>; the device prompts for a username and password.

Enter the password only. As some browsers require that the username be provided, it's recommended to enter the username with an arbitrary value.

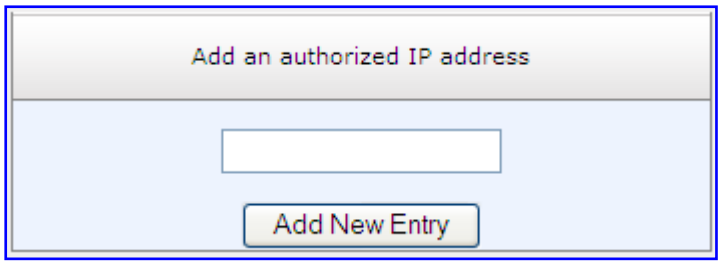
5.6 Configuring Web and Telnet Access List

The Web & Telnet Access List page is used to define IP addresses (up to ten) that are permitted to access the device's Web, Telnet, and SSH interfaces. Access from an undefined IP address is denied. If no IP addresses are defined, this security feature is inactive and the device can be accessed from any IP address. The Web and Telnet Access List can also be defined using the *ini* file parameter WebAccessList_x (see 'Web and Telnet Parameters' on page 290).

To add authorized IP addresses for Web, Telnet, and SSH interfaces access:

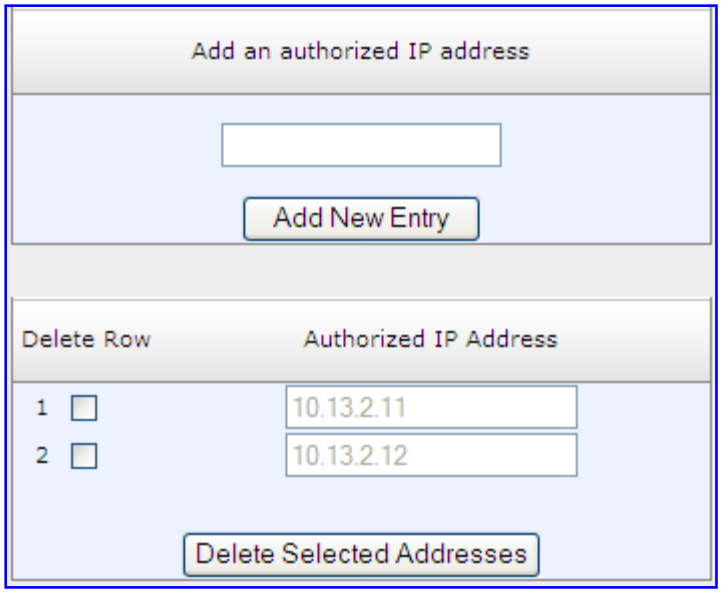
1. Open the Web & Telnet Access List page (**Configuration** tab >**System** menu >**Management** submenu >**Web & Telnet Access List**).

Figure 5-27: Web & Telnet Access List Page - Add New Entry



To add an authorized IP address, in the 'Add an authorized IP address' field, enter the required IP address, and then click **Add New Entry**; the IP address you entered is added as a new entry to the Web & Telnet Access List table.

Figure 5-28: Web & Telnet Access List Table



To delete authorized IP addresses, select the Delete Row check boxes corresponding to the IP addresses that you want to delete, and then click **Delete Selected Addresses**; the IP addresses are removed from the table and these IP addresses can no longer access the Web and Telnet interfaces.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

Notes:

The first authorized IP address in the list must be your PC's (terminal) IP address; otherwise, access from your PC is denied.

Delete your PC's IP address last from the 'Web & Telnet Access List page. If it is deleted before the last, subsequent access to the device from your PC is denied.

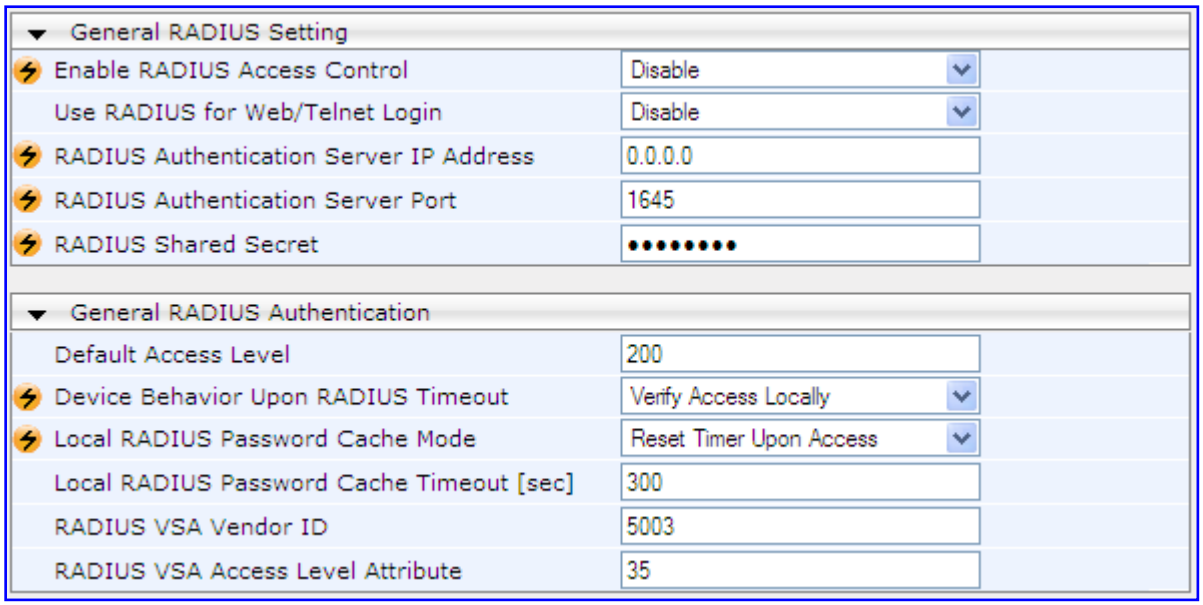
5.7 Configuring RADIUS Settings

The RADIUS Settings page is used for configuring the Remote Authentication Dial In User Service (RADIUS) accounting parameters. For a description of these parameters, see 'Configuration Parameters Reference' on page **Erreur ! Signet non défini.**.

To configure RADIUS:

1. Open the RADIUS Settings page (**Configuration** tab >**System** menu >**Management** submenu >**RADIUS Settings**).

Figure 5-29: RADIUS Parameters Page



Configure the parameters as required.

Click Subm**it to** apply your changes.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

6 CLI-Based Management

This section provides an overview of the CLI-based management and configuration relating to CLI management.

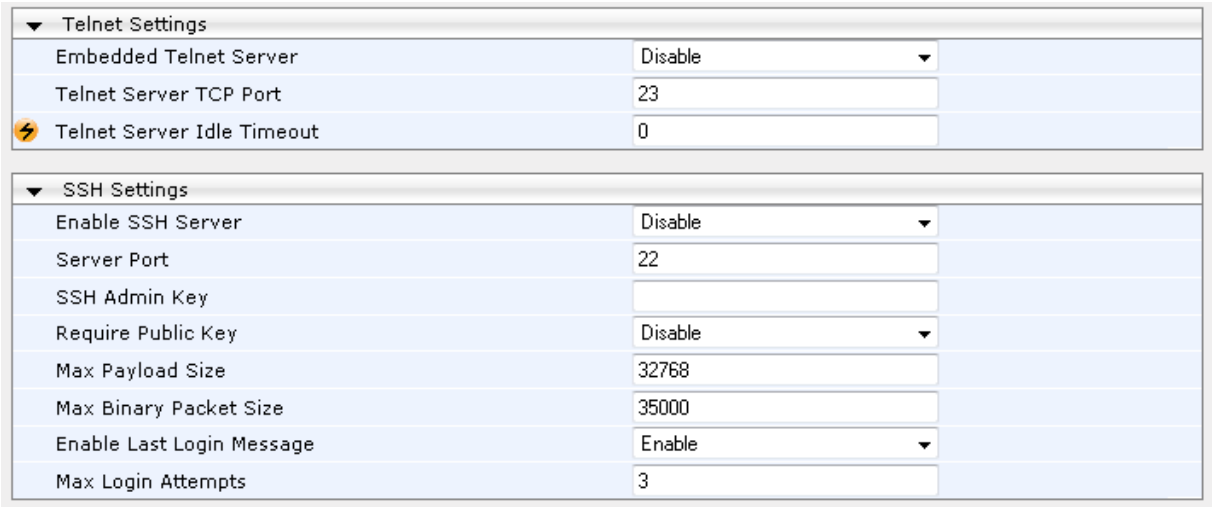
6.1 Configuring Telnet and SSH Settings

The Telnet/SSH Settings page is used to define Telnet and Secure Shell (SSH). For a description of these parameters, see 'Web and Telnet Parameters' on page 290.

To define Telnet and SSH:

1. Open the Telnet/SSH Settings page (**Configuration** tab >**System** menu >**Management** submenu >**Telnet/SSH Settings**).

Figure 6-1: Telnet/SSH Settings Page



Configure the parameters as required.

Click Subm**it to** apply your changes.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

7 SNMP-Based Management

The device provides an embedded SNMP Agent to operate with a third-party SNMP Manager for operation, administration, maintenance, and provisioning (OAMP) of the device. The SNMP Agent supports standard Management Information Base (MIBs) and proprietary MIBs, enabling a deeper probe into the interworking of the device. The SNMP Agent can also send unsolicited events (SNMP traps) towards the SNMP Manager. All supported MIB files are supplied to customers as part of the release.

This section provides configuration relating to SNMP management.

Note: For more information on SNMP support, refer to the *Product Reference Manual*.

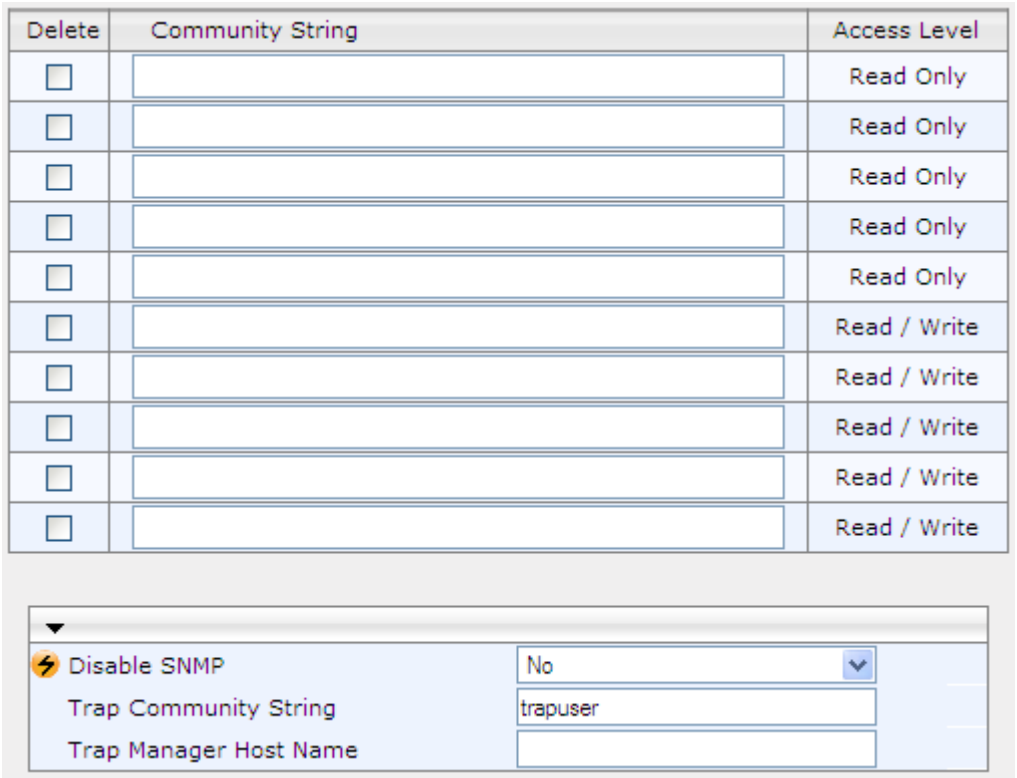
7.1 Configuring SNMP Community Strings

The SNMP Community String page allows you to configure up to five read-only and up to five read-write SNMP community strings, and to configure the community string that is used for sending traps. For more information on SNMP community strings, refer to the *Product Reference Manual*. For detailed descriptions of the SNMP parameters, see 'SNMP Parameters' on page 294.

To configure the SNMP community strings:

1. Open the SNMP Community String page (**Maintenance** tab >**System** menu >**Management** submenu >**SNMP** submenu >**SNMP Community String**).

Figure 7-1: SNMP Community String Page



Configure the SNMP community strings parameters according to the table below.

Click Submit to apply your changes.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

To delete a community string, select the **Delete** check box corresponding to the community string that you want to delete, and then click **Submit**.

Table 7-1: SNMP Community String Parameters Description

Parameter	Description
Community String	Read Only **[SNMPReadOnlyCommunityString_x]:** Up to five read-only community strings (up to 19 characters each). The default string is 'public'. Read / Write **[SNMPReadWriteCommunityString_x]:** Up to five read / write community strings (up to 19 characters each). The default string is 'private'.
Trap Community String **[SNMPTrapCommunityString]**	Community string used in traps (up to 19 characters). The default string is 'trapuser'.

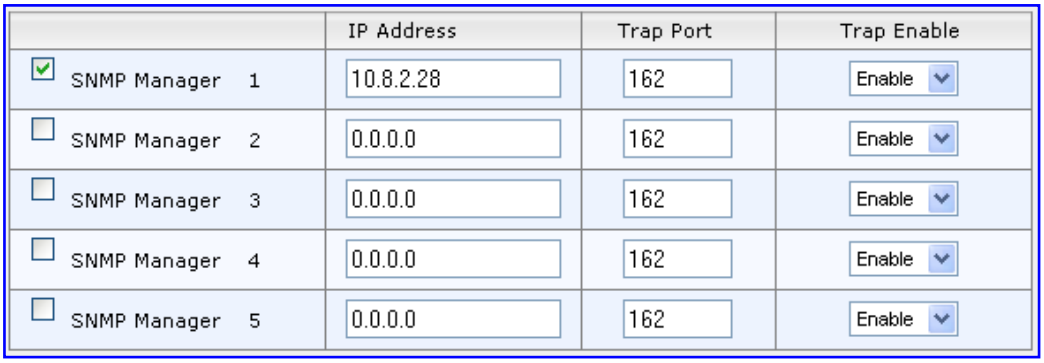
7.2 Configuring SNMP Trap Destinations

The SNMP Trap Destinations page allows you to configure up to five SNMP trap managers.

To configure SNMP trap destinations:

1. Open the SNMP Trap Destinations page (**Maintenance** tab >**System** menu >**Management** submenu >**SNMP** submenu >**SNMP Trap Destinations**).

Figure 7-2: SNMP Trap Destinations Page



Configure the SNMP trap manager parameters according to the table below.

Click Subm**it to** apply your changes.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

Note: Only table row entries whose corresponding check boxes are selected are applied when clicking **Submit**; otherwise, settings revert to their defaults.

Table 7-2: SNMP Trap Destinations Parameters Description

Parameter	Description
SNMP Manager **[SNMPManagerIsUsed_x]**	Determines the validity of the parameters (IP address and port number) of the corresponding SNMP Manager used to receive SNMP traps. **[0]** (Check box cleared) = Disabled (default) **[1]** (Check box selected) = Enabled
IP Address **[SNMPManagerTableIP_x]**	IP address of the remote host used as an SNMP Manager. The device sends SNMP traps to these IP addresses. Enter the IP address in dotted-decimal notation, e.g., 108.10.1.255.
Trap Port **[SNMPManagerTrapPort_x]**	Defines the port number of the remote SNMP Manager. The device sends SNMP traps to these ports. The valid SNMP trap port range is 100 to 4000. The default port is 162.
Trap Enable **[SNMPManagerTrapSendingEnable_x]**	Activates or de-activates the sending of traps to the corresponding SNMP Manager. **[0]** Disable = Sending is disabled. **[1]** Enable = Sending is enabled (default).

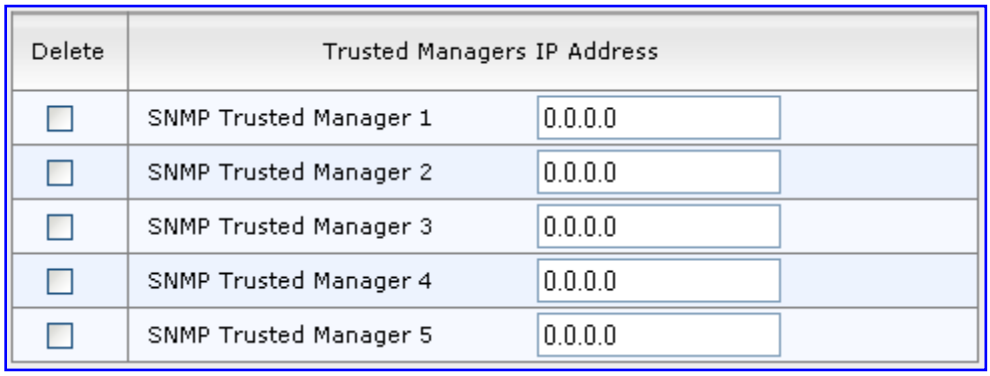
7.3 Configuring SNMP Trusted Managers

The SNMP Trusted Managers page allows you to configure up to five SNMP Trusted Managers, based on IP addresses. By default, the SNMP agent accepts SNMP Get and Set requests from any IP address, as long as the correct community string is used in the request. Security can be enhanced by using Trusted Managers, which is an IP address from which the SNMP agent accepts and processes SNMP requests.

To configure SNMP Trusted Managers:

1. Open the SNMP Trusted Managers page (**Maintenance** tab >**System** menu >**Management** submenu >**SNMP** submenu >**SNMP Trusted Managers**).

Figure 7-3: SNMP Trusted Managers



Select the check box corresponding to the SNMP Trusted Manager that you want to enable and for whom you want to define an IP address.

Define an IP address in dotted-decimal notation.

Click Subm**it to a**pply your changes.

To save the changes, see 'Saving Configuration' on page 245.

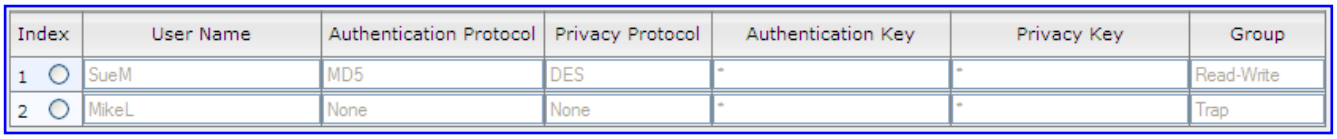
7.4 Configuring SNMP V3 Users

The SNMP v3 Users page allows you to configure authentication and privacy for up to 10 SNMP v3 users.

To configure the SNMP v3 users:

1. Open the SNMP v3 Users page (**Maintenance** tab >**System** menu >**Management** submenu >**SNMP** submenu >**SNMP V3 Users**).

Figure 7-4: SNMP V3 Setting Page



To add an SNMP v3 user, in the 'Add Index' field, enter the desired row index, and then click **Add Index**. A new row appears.

Configure the SNMP V3 Setting parameters according to the table below.

Click the **Apply** button to save your changes.

To save the changes, see 'Saving Configuration' on page 245.

Notes:

For a description of the web interface's table command buttons (e.g., **Duplicate** and **Delete**), see 'Working with Tables' on page 28.

You can also configure SNMP v3 users using the *ini* file table parameter SNMPUsers (see 'SNMP Parameters' on page 294).

Table 7-3: SNMP V3 Users Parameters

Parameter	Description
Index **[SNMPUsers_Index]**	The table index. The valid range is 0 to 9.
User Name **[SNMPUsers_Username]**	Name of the SNMP v3 user. This name must be unique.
Authentication Protocol **[SNMPUsers_AuthProtocol]**	Authentication protocol of the SNMP v3 user. **[0]** None (default) **[1]** MD5 **[2]** SHA-1
Privacy Protocol **[SNMPUsers_PrivProtocol]**	Privacy protocol of the SNMP v3 user. **[0]** None (default) **[1]** DES **[2]** 3DES **[3]** AES-128 **[4]** AES-192 **[5]** AES-256
Authentication Key **[SNMPUsers_AuthKey]**	Authentication key. Keys can be entered in the form of a text password or long hex string. Keys are always persisted as long hex strings and keys are localized.

Parameter	Description
Privacy Key **[SNMPUsers_PrivKey]**	Privacy key. Keys can be entered in the form of a text password or long hex string. Keys are always persisted as long hex strings and keys are localized.
Group **[SNMPUsers_Group]**	The group with which the SNMP v3 user is associated. **[0]** Read-Only (default) **[1]** Read-Write **[2]** Trap **Note:** All groups can be used to send traps.

8 INI File-Based Management

The *ini* file is a text-based file (created using, for example, Notepad) that can contain any number of parameters settings. The *ini* file can be loaded to the device using the following methods:

Web interface (see 'Backing Up and Loading Configuration File' on page 255)

Alcatel-Lucent' BootP/TFTP utility (refer to the Product Reference Manual)

Any standard TFTP server

When loaded to the device, the configuration settings of the *ini* file are saved to the device's non-volatile memory. If a parameter is excluded from the loaded *ini* file, the following occurs, depending on how you load the file:

Using the Load Auxiliary Files page (see 'Loading Auxiliary Files' on page 247): current settings are retained for excluded parameters

All other methods: default value is assigned to excluded parameters (according to the .cmp file running on the device), thereby, overriding values previously defined for these parameters

Notes:

For a list and description of the *ini* file parameters, see 'Configuration Parameters Reference' on page **Erreur ! Signet non défini.**.

Some parameters are configurable only through the *ini* file (and not the Web interface).

To restore the device to default settings using the *ini* file, see 'Restoring Factory Default Settings' on page 256.

8.1 INI File Format

The *ini* file can be configured with any number of parameters. These *ini* file parameters can be one of the following types:

Individual parameters (see 'Configuring Individual ini File Parameters' on page 53)

Table parameters (see 'Configuring ini File Table Parameters' on page 54)

8.1.1 Configuring Individual ini File Parameters

The format of individual *ini* file parameters includes an optional, subsection name (group name) to conveniently group similar parameters by their functionality. Following this line are the actual parameter settings. These format lines are shown below:

```
[subsection name]
; the subsection name is optional.
Parameter_Name = Parameter_Value
Parameter_Name = Parameter_Value
; Remark
; For example:
[System Parameters]
SyslogServerIP = 10.13.2.69
EnableSyslog = 1
; these are a few of the system-related parameters.
```

For general *ini* file formatting rules, see 'General ini File Formatting Rules' on page 55.

8.1.2 Configuring ini File Table Parameters

The *ini* file table parameters allow you to configure tables which can include multiple parameters (*columns*) and row entries (*indices*). When loading an *ini* file to the device, it's recommended to include only tables that belong to applications that are to be configured (dynamic tables of other applications are empty, but static tables are not).

The *ini* file table parameter is composed of the following elements:

Title of the table: The name of the table in square brackets (e.g., **[MY**_TABLE_NAME]).

Format line: Specifies the columns of the table (by their string names) that are to be configured.

The first word of the Format line must be 'FORMAT', followed by the Index field name and then an equal (=) sign. After the equal sign, the names of the columns are listed.

Columns must be separated by a comma (,).

The Format line must only include columns that can be modified (i.e., parameters that are not specified as read-only). An exception is Index fields, which are mandatory.

The Format line must end with a semicolon (;).

Data line(s): Contain the actual values of the columns (parameters). The values are interpreted according to the Format line.

The first word of the Data line must be the table's string name followed by the Index field.

Columns must be separated by a comma (,).

A Data line must end with a semicolon (;).

End-of-Table Mark: Indicates the end of the table. The same string used for the table's title, preceded by a backslash (\), e.g., **[\MY**_TABLE_NAME].

The following displays an example of the structure of an *ini* file table parameter.

```
[Table_Title]
; This is the title of the table.
FORMAT Index = Column_Name1, Column_Name2, Column_Name3;
; This is the Format line.
Index 0 = value1, value2, value3;
Index 1 = value1, $$, value3;
; These are the Data lines.
[\Table_Title]
; This is the end-of-the-table-mark.
```

The *ini* file table parameter formatting rules are listed below:

Indices (in both the Format and the Data lines) must appear in the same order. The Index field must never be omitted.

The Format line can include a subset of the configurable fields in a table. In this case, all other fields are assigned with the pre-defined default values for each configured line.

The order of the fields in the Format line isn't significant (as opposed to the Index fields). The fields in the Data lines are interpreted according to the order specified in the Format line.

The double dollar sign ($$) in a Data line indicates the default value for the parameter.

The order of the Data lines is insignificant.

Data lines must match the Format line, i.e., it must contain exactly the same number of Indices and Data fields and must be in exactly the same order.

A row in a table is identified by its table name and Index field. Each such row may appear only once in the *ini* file.

Table dependencies: Certain tables may depend on other tables. For example, one table may include a field that specifies an entry in another table. This method is used to specify

additional attributes of an entity, or to specify that a given entity is part of a larger entity. The tables must appear in the order of their dependency (i.e., if Table X is referred to by Table Y, Table X must appear in the *ini* file before Table Y).

For general *ini* file formatting rules, see 'General ini File Formatting Rules' on page 55.

The table below displays an example of an *ini* file table parameter:

```
[AccessList]
FORMAT AccessList_Index = AccessList_Source_IP,
AccessList_Source_Port, AccessList_PrefixLen, AccessList_Source_Port,
AccessList_Start_Port, AccessList_End_Port, AccessList_Protocol,
AccessList_Use_Specific_Interface, AccessList_Interface_ID,
AccessList_Packet_Size, AccessList_Byte_Rate, AccessList_Byte_Burst,
AccessList_Allow_Type;
[\AccessList]
```

Note: Do not include read-only parameters in the *ini* file table parameter as this can cause an error when attempting to load the file to the device.

8.1.3 General ini File Formatting Rules

The *ini* file must adhere to the following formatting rules:

The *ini* file name must not include hyphens (-) or spaces; if necessary, use an underscore (_) instead.

Lines beginning with a semi-colon (;) are ignored. These can be used for adding remarks in the *ini* file.

A carriage return (i.e., Enter) must be done at the end of each line.

The number of spaces before and after the equals sign (=) is irrelevant.

Subsection names for grouping parameters are optional.

If there is a syntax error in the parameter name, the value is ignored.

Syntax errors in the parameter's value can cause unexpected errors (parameters may be set to the incorrect values).

Parameter string values that denote file names (e.g., CallProgressTonesFileName) must be enclosed with inverted commas ('…'), e.g., CallProgressTonesFileName = 'cpt_usa.dat'

The parameter name is not case-sensitive.

The parameter value is not case-sensitive, except for coder names.

The *ini* file must end with at least one carriage return.

8.2 Modifying an ini File

You can modify an *ini* file currently used by the device. Modifying an *ini* file instead of loading an entirely new *ini* file preserves the device's current configuration.

To modify an *ini* file:

1. Save the current *ini* file from the device to your PC, using the Web interface (see 'Backing Up and Loading Configuration File' on page 255).

Open the *ini* file (using a text file editor such as Notepad), and then modify the *ini* file parameters according to your requirements.

Save the modified *ini* file, and then close the file.

Load the modified *ini* file to the device, using the Web interface (see 'Backing Up and Loading Configuration File' on page 255).

Tip: Before loading the *ini* file to the device, verify that the file extension of the *ini* file is correct, i.e., *.ini*.

8.3 Secured Encoded ini File

The *ini* file contains sensitive information that is required for the functioning of the device. The file may be loaded to the device using HTTP. These protocols are not secure and are vulnerable to potential hackers. To overcome this security threat, the Alcatel-Lucent' TrunkPack Downloadable Conversion Utility (DConvert) utility allows you to binary-encode (encrypt) the *ini* file before loading it to the device (refer to the *Product Reference Manual)*.

Notes:

The procedure for loading an encoded *ini* file is identical to the procedure for loading an unencoded *ini* file (see 'Backing Up and Loading Configuration File' on page 255).

If you download from the device (to a folder on your PC) an *ini* file that was loaded encoded to the device, the file is saved as a regular *ini* file (i.e., unencoded).

Part III

General System Settings

This part provides general system configurations.

9 Configuring Certificates

The Certificates page is used for configuring secure communication using HTTPS and SIP TLS. This page allows you to do the following:

Replace the device's certificate - see 'Replacing Device Certificate' on page 58

Load a new private key from an external source - see 'Loading a Private Key' on page 61

Configure trusted root certificates - see 'Mutual TLS Authentication' on page 62

Regenerate keys and self-signed certificates - see 'Self-Signed Certificates' on page 63

Note: The device is shipped with a working TLS configuration. Therefore, configure certificates only if required.

9.1 Replacing Device Certificate

The device is supplied with a working Transport Layer Security (TLS) configuration consisting of a unique self-signed server certificate. If an organizational Public Key Infrastructure (PKI) is used, you may wish to replace this certificate with one provided by your security administrator.

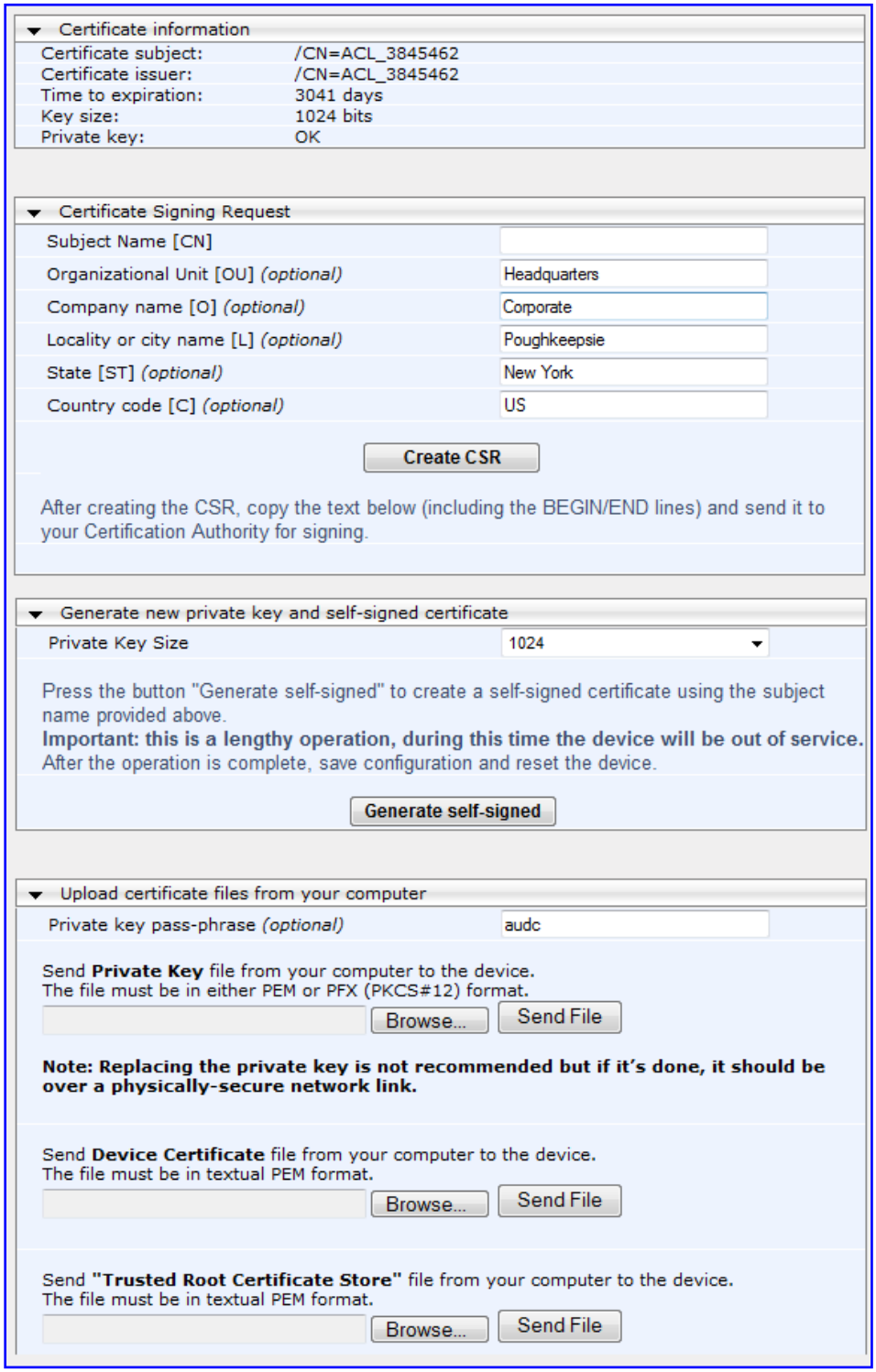
To replace the device's certificate:

1. Your network administrator should allocate a unique DNS name for the device (e.g., dns_name.corp.customer.com). This DNS name is used to access the device and therefore, must be listed in the server certificate.

If the device is operating in HTTPS mode, then set the 'Secured Web Connection (HTTPS)' field (HTTPSOnly) to **HTTP and HTTPS** (see 'Configuring Web Security Settings' on page 41). This ensures that you have a method for accessing the device in case the new certificate does not work. Restore the previous setting after testing the configuration.

Open the Certificates page (**Configuration** tab >**System** menu >**Certificates**).

Figure 9-1: Certificates Page


Certificate information
Certificate subject: /CN=ACL_3845462
Certificate issuer: /CN=ACL_3845462
Time to expiration: 3041 days
Key size: 1024 bits
Private key: OK
Certificate Signing Request
Subject Name [CN]
Organizational Unit [OU] (optional) Headquarters
Company name [O] (optional) Corporate
Locality or city name [L] (optional) Poughkeepsie
State [ST] (optional) New York
Country code [C] (optional) US
Create CSR
After creating the CSR, copy the text below (including the BEGIN/END lines) and send it to your Certification Authority for signing.
Generate new private key and self-signed certificate
Private Key Size 1024
Press the button "Generate self-signed" to create a self-signed certificate using the subject name provided above.
Important: this is a lengthy operation, during this time the device will be out of service.
After the operation is complete, save configuration and reset the device.
Generate self-signed
Upload certificate files from your computer
Private key pass-phrase (optional) audc
Send Private Key file from your computer to the device.
The file must be in either PEM or PFX (PKCS#12) format.
Browse... Send File
Note: Replacing the private key is not recommended but if it's done, it should be over a physically-secure network link.
Send Device Certificate file from your computer to the device.
The file must be in textual PEM format.
Browse... Send File
Send "Trusted Root Certificate Store" file from your computer to the device.
The file must be in textual PEM format.
Browse... Send File

Under the **Certificate Signing Request** group, do the following:

a. In the 'Subject Name **[CN]**' field, enter the DNS name.

Fill in the rest of the request fields according to your security provider's instructions.

Click **CreateCSR**; a textual certificate signing request is displayed.

Copy the text and send it to your security provider. The security provider (also known as Certification Authority or CA) signs this request and then sends you a server certificate for the device.

Save the certificate to a file (e.g., cert.txt). Ensure that the file is a plain-text file containing the 'BEGIN CERTIFICATE' header, as shown in the example of a Base64-Encoded X.509 Certificate below:

```
-----BEGIN CERTIFICATE-----
MIIDkzCCAnugAwIBAgIEAgAAADANBgkqhkiG9w0BAQQFADA/MQswCQYDVQQGEwJGUjETM
BEGA1UEChMKQ2VydGlwb3N0ZTEbMBkGA1UEAxMSQ2VydGlwb3N0ZSBTZXJ2ZXVyMB4XDT
k4MDYyNDA4MDAwMFoXDTE4MDYyNDA4MDAwMFowPzELMAkGA1UEBhMCRlIxEzARBgNVBAo
TCkNlcnRpcG9zdGUxGzAZBgNVBAMTEkNlcnRpcG9zdGUgU2VydmV1cjCCASEwDQYJKoZI
hvcNAQEBBQADggEOADCCAQkCggEAPqd4MziR4spWldGRx8bQrhZkonWnNm`+Yhb7+4Q67
ecf1janH7GcN/SXsfx7jJpreWULf7v7Cvpr4R7qIJcmdHIntmf7JPM5n6cDBv17uSW63e
r7NkVnMFHwK1QaGFLMybFkzaeGrvFm4k3lRefiXDmuOe+FhJgHYezYHf44LvPRPwhSrzi
9+Aq3o8pWDguJuZDIUP1F1jMa+LPwvREXfFcUW+w==
-----END CERTIFICATE-----
```

Scroll down to the **Upload certificates files from your computer** group, click the **Browse** button corresponding to the 'Send Device Certificate...' field, navigate to the cert.txt file, and then click **Send File**.

After the certificate successfully loads to the device, save the configuration with a device reset (see 'Saving Configuration' on page 245); the Web interface uses the provided certificate.

Open the Certificates page again and verify that under the **Certificate information** group (at the top of the page), the 'Private key' read-only field displays "OK"; otherwise, consult your security administrator.

If the device was originally operating in HTTPS mode and you disabled it in Step 2, then return it to HTTPS by setting the 'Secured Web Connection (HTTPS)' field to **HTTPS Only**.

Notes:

The certificate replacement process can be repeated when necessary (e.g., the new certificate expires).

It is possible to use the IP address of the device (e.g., 10.3.3.1) instead of a qualified DNS name in the Subject Name. This is not recommended since the IP address is subject to changes and may not uniquely identify the device.

The device certificate can also be loaded via the Automatic Update Facility, using the HTTPSCertFileName *ini* file parameter.

9.2 Loading a Private Key

The device is shipped with a self-generated random private key, which cannot be extracted from the device. However, some security administrators require that the private key be generated externally at a secure facility and then loaded to the device through configuration. Since private keys are sensitive security parameters, take precautions to load them over a physically-secure connection such as a back-to-back Ethernet cable connected directly to the managing computer.

To replace the device's private key:

1. Your security administrator should provide you with a private key in either textual PEM (PKCS #7) or PFX (PKCS #12) format. The file may be encrypted with a short pass-phrase, which should be provided by your security administrator.

If the device is operating in HTTPS mode, then set the 'Secured Web Connection (HTTPS)' field (HTTPSOnly) to **HTTP and HTTPS** (see 'Configuring Web Security Settings' on page 41). This ensures that you have a method for accessing the device in case the new configuration does not work. Restore the previous setting after testing the configuration.

Open the Certificates page (**Configuration** tab >**System** menu >**Certificates**) and scroll down to the **Upload certificate files from your computer** group.

Fill in the 'Private key pass-phrase' field, if required.

Click the **Browse** button corresponding to the 'Send Private Key' field, navigate to the key file, and then click **Send File**.

If the security administrator has provided you with a device certificate file, load it using the 'Send Device Certificate' field.

After the files successfully load to the device, save the configuration with a device reset (see 'Saving Configuration' on page 245); the Web interface uses the new configuration.

Open the Certificates page again, and verify that under the **Certificate information** group (at the top of the page) the 'Private key' read-only field displays "OK"; otherwise, consult your security administrator.

If the device was originally operating in HTTPS mode and you disabled it in Step 2, then enable it by setting the 'Secured Web Connection (HTTPS)' field to **HTTPS Only**.

9.3 Mutual TLS Authentication

By default, servers using TLS provide one-way authentication. The client is certain that the identity of the server is authentic. When an organizational PKI is used, two-way authentication may be desired - both client and server should be authenticated using X.509 certificates. This is achieved by installing a client certificate on the managing PC and loading the root CA's certificate to the device's Trusted Root Certificate Store. The Trusted Root Certificate file may contain more than one CA certificate combined, using a text editor.

Since X.509 certificates have an expiration date and time, the device must be configured to use NTP (see 'Simple Network Time Protocol Support' on page 65) to obtain the current date and time. Without the correct date and time, client certificates cannot work.

To enable mutual TLS authentication for HTTPS:

1. Set the 'Secured Web Connection (HTTPS)' field to **HTTPS Only** (see 'Configuring Web Security Settings' on page 41) to ensure you have a method for accessing the device in case the client certificate does not work. Restore the previous setting after testing the configuration.

Open the Certificates page (see 'Replacing Device Certificate' on page 58).

In the **Upload certificate files from your computer** group, click the **Browse** button corresponding to the 'Send Trusted Root Certificate Store ...' field, navigate to the file, and then click **Send File**.

When the operation is complete, set the 'Requires Client Certificates for HTTPS connection' field to **Enable** (see 'Configuring Web Security Settings' on page 41).

Save the configuration with a device reset (see 'Saving Configuration' on page 245).

When a user connects to the secured Web interface of the device:

If the user has a client certificate from a CA that is listed in the Trusted Root Certificate file, the connection is accepted and the user is prompted for the system password.

If both the CA certificate and the client certificate appear in the Trusted Root Certificate file, the user is not prompted for a password (thus, providing a single-sign-on experience - the authentication is performed using the X.509 digital signature).

If the user does not have a client certificate from a listed CA or does not have a client certificate, the connection is rejected.

Notes:

The process of installing a client certificate on your PC is beyond the scope of this document. For more information, refer to your operating system documentation, and/or consult your security administrator.

The root certificate can also be loaded via the Automatic Update facility, using the HTTPSRootFileName *ini* file parameter.

You can enable Online Certificate Status Protocol (OCSP) on the device to check whether a peer's certificate has been revoked by an OCSP server. For more information, refer to the *Product Reference Manual*.

9.4 Self-Signed Certificates

The device is shipped with an operational, self-signed server certificate. The subject name for this default certificate is 'ACL_nnnnnnn', where *nnnnnnn* denotes the serial number of the device. However, this subject name may not be appropriate for production and can be changed while still using self-signed certificates.

To change the subject name and regenerate the self-signed certificate:

1. Before you begin, ensure the following:

   You have a unique DNS name for the device (e.g., dns_name.corp.customer.com). This name is used to access the device and should therefore, be listed in the server certificate.

   No traffic is running on the device. The certificate generation process is disruptive to traffic and should be executed during maintenance time.

Open the Certificates page (see 'Replacing Device Certificate' on page 58).

In the 'Subject Name **[CN]**' field, enter the fully-qualified DNS name (FQDN) as the certificate subject, select the desired private key size (in bits), and then click **Generate self-signed**; after a few seconds, a message appears displaying the new subject name.

Save the configuration with a device reset (see 'Saving Configuration' on page 245) for the new certificate to take effect.

10 Date and Time

The date and time of the device can be configured manually or it can be obtained automatically from a Simple Network Time Protocol (SNTP) server.

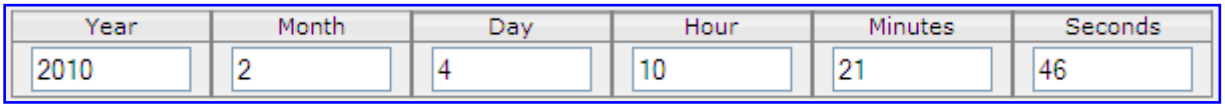
10.1 Manual Date and Time

The date and time of the device can be configured manually.The Regional Settings page allows you to define and view the device's internal date and time.

To configure the device's date and time:

1. Open the Regional Settings page (**Configuration** tab >**System** menu >**Regional Settings**).

Figure 10-1: Regional Settings Page



Enter the current date and time in the geographical location in which the device is installed.

Click the **Submit** button; the date and time are automatically updated.

Notes:

If the device is configured to obtain the date and time from an Simple Network Time Protocol Support (SNTP) server, the fields on this page display the received date and time and are read-only.

After performing a hardware reset, the date and time are returned to their defaults and therefore, should be updated.

10.2 Automatic Date and Time through SNTP Server

The Simple Network Time Protocol (SNTP) client functionality generates requests and reacts to the resulting responses using the NTP version 3 protocol definitions (according to RFC 1305). Through these requests and responses, the NTP client synchronizes the system time to a time source within the network, thereby eliminating any potential issues should the local system clock 'drift' during operation. By synchronizing time to a network time source, traffic handling, maintenance, and debugging become simplified for the network administrator.

The NTP client follows a simple process in managing system time: the NTP client requests an NTP update, receives an NTP response, and then updates the local system clock based on a configured NTP server within the network.

The client requests a time update from a specified NTP server at a specified update interval. In most situations, this update interval is every 24 hours based on when the system was restarted. The NTP server identity (as an IP address) and the update interval are user-defined (using the *ini* file parameters NTPServerIP and NTPUpdateInterval respectively), or an SNMP MIB object (refer to the *Product Reference Manual*).

When the client receives a response to its request from the identified NTP server, it must be interpreted based on time zone or location offset that the system is to a standard point of reference called the Universal Time Coordinate (UTC). The time offset that the NTP client uses is configurable using the *ini* file parameter NTPServerUTCOffset, or via an SNMP MIB object (refer to the *Product Reference Manual*).

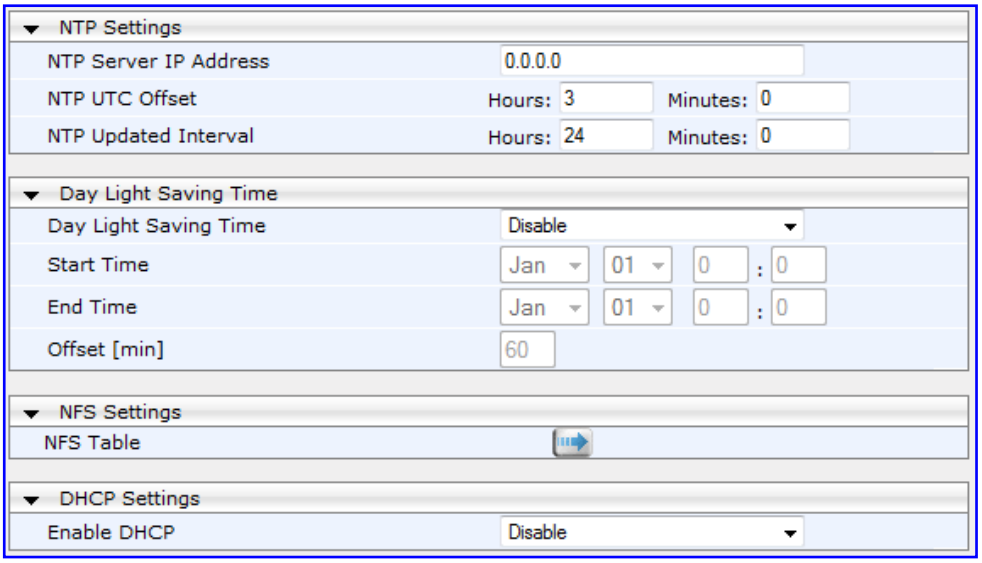
If required, the clock update is performed by the client as the final step of the update process. The update is performed in such a way as to be transparent to the end users. For instance, the response of the server may indicate that the clock is running too fast on the client. The client slowly robs bits from the clock counter to update the clock to the correct time. If the clock is running too slow, then in an effort to catch the clock up, bits are added to the counter, causing the clock to update quicker and catch up to the correct time. The advantage of this method is that it does not introduce any disparity in the system time that is noticeable to an end user or that could corrupt call timeouts and timestamps.

The procedure below describes how to configure SNTP using the Web interface.

To configure SNTP using the Web interface:

1. Open the Application Settings page (**Configuration** tab >**System** menu >**Application Settings**).

Figure 10-2: Applications Settings Page



Configure the NTP parameters:

'NTP Server IP Address' (NTPServerIP) - defines the IP address of the NTP server

'NTP UTC Offset' (NTPServerUTCOffset) - defines the time offset in relation to the UTC. For example, if your region is 2 hours ahead of the UTC, enter "2".

'NTP Updated Interval' (NTPUpdateInterval) - defines the period after which the date and time of the device is updated

Configure daylight saving, if required:

'Day Light Saving Time' (DayLightSavingTimeEnable) - enables daylight saving time

'Start Time' (DayLightSavingTimeStart) and 'End Time' (DayLightSavingTimeEnd) - defines the period for which daylight saving time is relevant.

'Offset' (DayLightSavingTimeOffset) - defines the offset in minutes to add to the time for daylight saving. For example, if your region has daylight saving of one hour, the time received from the NTP server is 11:00, and the UTC offset for your region is +2 (i.e., 13:00), you need to enter "60" to change the local time to 14:00.

Verify that the device is set to the correct date and time. You can do this by viewing the date and time in the Regional Settings page, as described in 'Configuring Date and Time' on page 64.

Part IV

VoIP Configuration

This part describes VoIP configuration.

11 Network

This section describes the network-related configuration.

11.1 Configuring Physical Ethernet Ports

The physical LAN ports are each assigned a group number. Each group can then be assigned to IP network interfaces in the Multiple Interface table (see 'Configuring IP Interface Settings' on page 69), thereby allowing physical separation of network interfaces. By the means of physical separation of interfaces, the administrator can gain higher level of segregation of sub-networks. Equipment connected to different physical ports is not accessible to one other. The only connection between them can be established by cross connecting them with media stream (a VoIP call).

For each LAN port, you can configure the speed, duplex mode, native VLAN (PVID), and provide a brief description.

The Ethernet ports can also be configured using the ini file parameter table, PhysicalPortsTable.

To configure the physical Ethernet ports:

2. Open the Physical Ports Settings page (**Configuration** tab >**VoIP** menu >**Network** submenu >**Physical Ports Settings**).

Figure 11-1: Physical Ports Settings Page

Index	Port	Mode	Native Vlan	Speed&Duplex	Description	Group Member	Group Status
1	GE_1	Enable	1	Auto Negotiation	User Port #0	GROUP_1	
2	GE_2	Enable	1	Auto Negotiation	User Port #1	GROUP_2	

Select the 'Index' radio button corresponding to the port that you want to configure.

Click the **Edit** button.

Configure the ports (see the table below for a description of the parameters).

Click **Apply** and then **Done**.

Physical Port Settings Parameters Description

Parameter	Description
Port	(Read-only) Displays the port number. The displayed string value represents the corresponding physical port on your server.
Mode	(Read-only) Displays the mode of the port: [0] Disable [1] Enable (default)
Native Vlan	Defines the Native VLAN or PVID of the port. Incoming packets without a VLAN ID are tagged with this VLAN. For outgoing packets, if the VLAN ID as defined in the Multiple Interface table is the same as the Native VLAN ID, the device sends the packet without a VLAN; otherwise, the VLAN ID as defined in the Multiple Interface table takes precedence. The valid value range is 1 to 4096. The default is 1.

Parameter	Description
Speed & Duplex	Defines the speed and duplex mode of the port. [0] 10BaseT Half Duplex [1] 10BaseT Full Duplex [2] 100BaseT Half Duplex [3] 100BaseT Full Duplex [4] Auto Negotiation (default) [6] 1000BaseT Half Duplex [7] 1000BaseT Full Duplex
Description	Defines an arbitrary description of the port.
Group Member	(Read-only) Displays the group to which the port belongs.
Group Status	This field is reserved for future use.

11.2 Configuring IP Interface Settings

The Multiple Interface Table page allows you to configure logical VoIP network interfaces. Each interface can be defined with the following:

Application type allowed on the interface:

- Control - call control signaling traffic (i.e., SIP)
- Media - RTP traffic
- Operations, Administration, Maintenance and Provisioning (OAMP) - management (such as Web- and SNMP-based management)
- Maintenance - Maintenance interface used in High Availability (HA) mode - this interface represents one of the LAN interfaces or Ethernet groups on each device used for the Ethernet connectivity between the two devices

IP address and subnet

VLAN ID

Default Gateway

Primary and secondary DNS IP address

Associated physical Ethernet port group (Underlying Device) used for the interface - useful for setting trusted and un-trusted networks on different physical ports

You can configure up to 48 interfaces - up to 47Control and Media interfaces (including a Maintenance interface if implementing an HA system) and 1 OAMP interface. A combination of multiple IP addresses of IPv4 and IPv6 interfaces can also be defined. However, only one interface (of IPv4 type) must be defined for OAMP; the rest being Media, Control, or a combination of Media and Control. The IPv6 Internet Layer protocol is based on the definition of a 128-bit address (as opposed to 32 bits for IPv4).

The default VoIP interface is as follows:

Application type: OAMP + Media + Control

IP address: 192.168.0.2 with prefix length 24 (i.e., subnet mask 255.255.255.0)

Default gateway: 192.168.0.1

Name: "Voice"

VLAN ID: 1

Notes:

For more information on HA and configuring the HA Maintenance application type, see 'HA Configuration' on page 233.

For more information and examples of VoIP network interfaces configuration, see 'Network Configuration' on page 73.

You can define firewall rules (access list) to deny (block) or permit (allow) packets received from a specific IP interface configured in this table. These rules are configured using the AccessList parameter (see 'Configuring the Access List' on page 98).

You can view currently active configured IP interfaces in the 'IP Active Interfaces page (see 'Viewing Active IP Interfaces' on page 269).

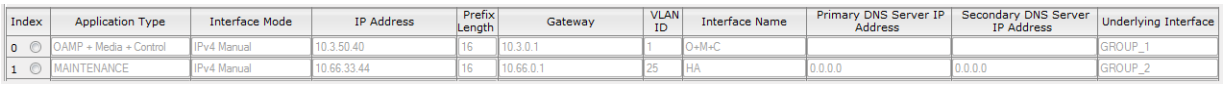
You can also configure this table using the *ini* file table parameter InterfaceTable (see 'Networking Parameters' on page 280).

For configuring Web interface tables, see 'Working with Tables' on page 28.

To configure VoIP network interfaces:

1. Open the Multiple Interface Table page (**Configuration** tab >**VoIP** menu >**Network** submenu >**IP Settings**).

Figure 11-2: Multiple Interface Table Page



Index	Application Type	Interface Mode	IP Address	Prefix Length	Gateway	VLAN ID	Interface Name	Primary DNS Server IP Address	Secondary DNS Server IP Address	Underlying Interface
0	OAMP + Media + Control	IPv4 Manual	10.3.50.40	16	10.3.0.1	1	O+M+C			GROUP_1
1	MAINTENANCE	IPv4 Manual	10.66.33.44	16	10.66.0.1	25	HA	0.0.0.0	0.0.0.0	GROUP_2

In the 'Add Index' field, enter the desired index number for the new interface, and then click **Add Index**; the index row is added to the table.

Configure the interface according to the table below.

Click the **Apply** button; the interface is added to the table and the **Done** button appears.

Click **Done** to validate the interface. If the interface is not valid (e.g., if it overlaps with another interface in the table or if it does not adhere to the other rules as summarized in 'Multiple Interface Table Configuration Summary and Guidelines' on page 78), a warning message is displayed.

Save the changes to flash memory and reset the device (see 'Saving Configuration' on page 245).

To view network interfaces that are currently active, click the **IP Interface Status Table** button. For a description of this display, see 'Viewing Active IP Interfaces' on page 269.

Table 11-1: Multiple Interface Table Parameters Description

Parameter	Description
Table parameters	
Index	Table index row of the interface. The range is 0 to 47.
Web: Application Type EMS: Application Types **[InterfaceTable_ApplicationTypes]**	Types of applications that are allowed on the specific interface. **[0]** OAMP = Only Operations, Administration, Maintenance and Provisioning (OAMP) applications (e.g., Web, Telnet, SSH, and SNMP) are allowed on the interface. **[1]** Media = Only Media (i.e., RTP streams of voice) is allowed on the interface. **[2]** Control = Only Call Control applications (e.g., SIP) are allowed on the interface. **[3]** OAMP + Media = Only OAMP and Media applications are

Parameter	Description
	allowed on the interface. **[4]** OAMP + Control = Only OAMP and Call Control applications are allowed on the interface. **[5]** Media + Control = Only Media and Call Control applications are allowed on the interface. **[6]** OAMP + Media + Control = All application types are allowed on the interface. **[99]** MAINTENANCE = Only the HA Maintenance application is allowed on this interface **Note:**For valid configuration guidelines, see 'Multiple Interface Table Configuration Summary and Guidelines' on page 78.
Web: Interface Mode **[InterfaceTable_InterfaceMode]**	Determines the method that this interface uses to calculate its IP address. **[3]** IPv6 Manual Prefix = IPv6 manual prefix IP address assignment. **[4]** IPv6 Manual = IPv6 manual IP address assignment. **[10]** IPv4 Manual = IPv4 manual IP address assignment.
Web/EMS: Prefix Length **[InterfaceTable_PrefixLength]**	Defines the Classless Inter-Domain Routing (CIDR)-style representation of a dotted decimal subnet notation. The CIDR-style representation uses a suffix indicating the number of bits which are set in the dotted decimal format (e.g. 192.168.0.0/16 is synonymous with 192.168.0.0 and a subnet of 255.255.0.0. Defines the number of '1' bits in the subnet mask (i.e., replaces the standard dotted-decimal representation of the subnet mask for IPv4 interfaces). For example: A subnet mask of 255.0.0.0 is represented by a prefix length of 8 (i.e., 11111111 00000000 00000000 00000000), and a subnet mask of 255.255.255.252 is represented by a prefix length of 30 (i.e., 11111111 11111111 11111111 11111100). The prefix length is a Classless Inter-Domain Routing (CIDR) style presentation of a dotted-decimal subnet notation. The CIDR-style presentation is the latest method for interpretation of IP addresses. Specifically, instead of using eight-bit address blocks, it uses the variable-length subnet masking technique to allow allocation on arbitrary-length prefixes (refer to http://en.wikipedia.org/wiki/Classless_Inter-Domain_Routing for more information). For IPv4 Interfaces, the prefix length values range from 0 to 31.For IPv6 interfaces, the prefix length must be set to 64. **Note:** Subnets of different interfaces must not overlap in any way (e.g., defining two interfaces with 10.0.0.1/8 and 10.50.10.1/24 is invalid). Each interface must have its own address space.
Web/EMS: Gateway **[InterfaceTable_Gateway]**	Defines the IP address of the default gateway for this interface. **Notes:** A default gateway can be defined for each interface. The default gateway's IP address must be in the same subnet as the interface address.
Web/EMS: VLAN ID **[InterfaceTable_VlanID]**	Defines the VLAN ID for each interface. **Note:** The VLAN ID must be unique for each interface.
Web/EMS: Interface Name	Defines a string (up to 16 characters) to name this interface. This name is displayed in management interfaces (Web, CLI and

Parameter	Description
[InterfaceTable_InterfaceName]	SNMP) for clarity (and has no functional use), as well as in the Media Realm table and SIP Interface table. **Notes:** This parameter is mandatory. The name must be unique for each interface.
Web/EMS: Primary DNS Server IP address **[InterfaceTable_PrimaryDNSServerIPAddress]**	Defines the IP address (in dotted-decimal notation) of the primary DNS server that is used for translating domain names into IP addresses for each interface. **Note:** This parameter is optional.
Web/EMS: Secondary DNS Server IP address **[InterfaceTable_SecondaryDNSServerIPAddress]**	Defines the IP address (in dotted-decimal notation) of the secondary DNS server that is used for translating domain names into IP addresses for each interface. **Note:** This parameter is optional.
Underlying Interface **[InterfaceTable_UnderlyingInterface]**	Assigns a physical Ethernet port(Group Member) to the IP interface. This is useful for separating trusted networks from un-trusted networks, by assigning each to different physical ports. To view the port groups and configure port settings, see "Configuring Physical Ethernet Ports" on page 68.

11.2.1 Network Configuration Overview

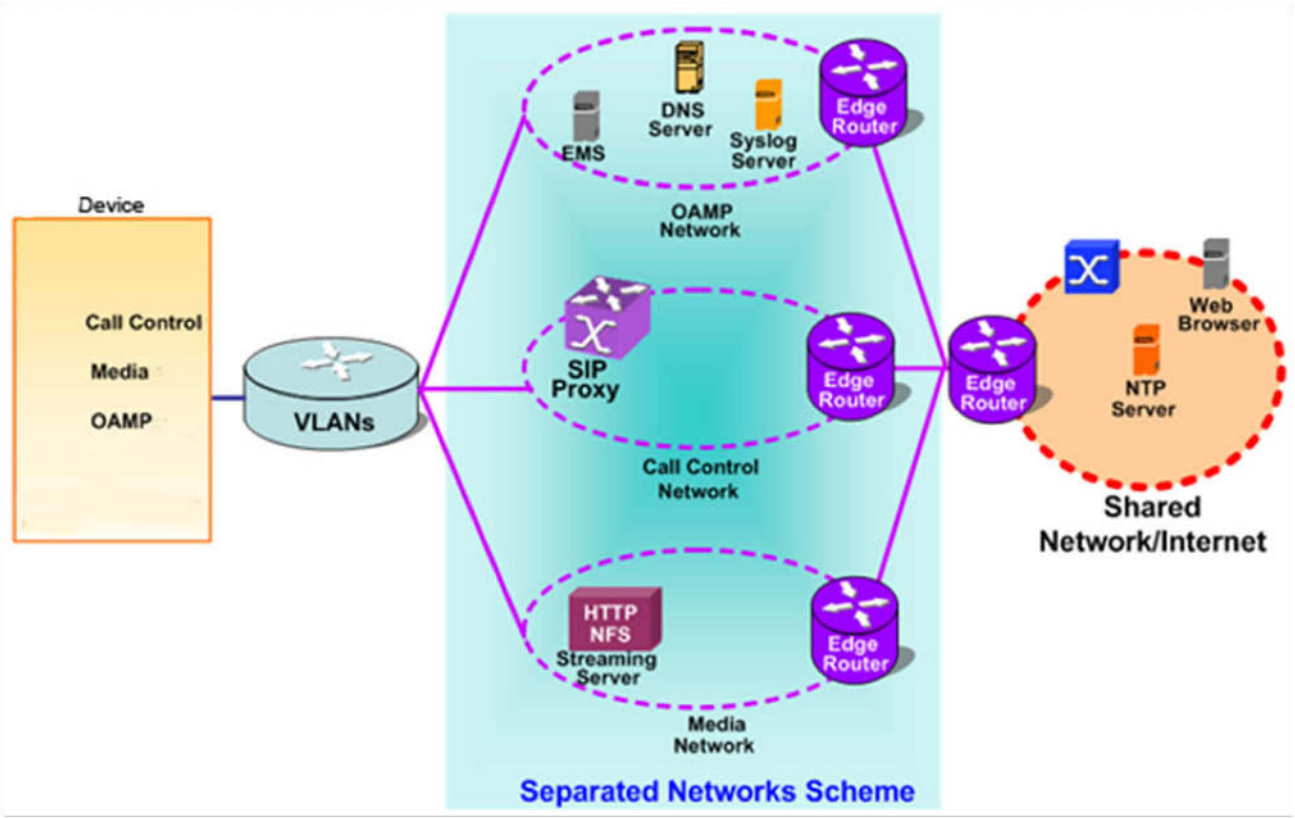
The device allows you to configure multiple IP addresses with associated VLANs for the VoIP network, using the Multiple Interface table. Complementing this table is the Routing table, which allows you to define VoIP network static routing rules for non-local hosts/subnets. This section describes the various network configuration options offered by the device.

This section covers the VoIP network configuration (interfaces, static routing rules, and QoS definitions).

11.2.1.1 Multiple Network Interfaces and VLANs

A need often arises to have logically separated network segments for various applications (for administrative and security reasons). This can be achieved by employing Layer-2 VLANs and Layer-3 subnets.

Figure 11-3: Multiple Network Interfaces



The figure depicts a typical configuration featuring in which the device is configured with three network interfaces for:

Operations, Administration, Maintenance, and Provisioning (OAMP) applications

Call Control applications

Media

The Multiple Interfaces scheme allows the configuration of different IP addresses, each associated with a unique VLAN ID. The configuration is performed using the Multiple Interface table, which is configurable using the *ini* file, Web, and SNMP interfaces.

11.2.1.1.1 Overview of Multiple Interface Table

The Multiple Interfaces scheme allows you to define different IP addresses and VLANs in a table format, as shown below:

Table 11-2: Multiple Interface Table

Index Mode	Application	Interface	IP Address	Prefix Length	Default Gateway	VLAN ID	Interface Name
0	OAMP	IPv4	10.31.174.50	16	0.0.0.0	4	ManagementIF
1	Control	IPv4	10.32.174.50	16	0.0.0.0	5	ControlIF
2	Media	IPv4	10.33.174.50	16	10.33.0.1	6	Media1IF
3	Media	IPv4	10.34.174.50	16	0.0.0.0	7	Media2IF
4	Media	IPv4	10.35.174.50	16	10.35.0.1	8	Media3IF
5	Media	IPv4	10.36.174.50	16	0.0.0.0	9	Media4IF
6	Media	IPv4	10.37.174.50	16	0.0.0.0	10	Media5IF
7	Media	IPv4	10.38.174.50	16	0.0.0.0	11	Media6IF
8	Media	IPv4	10.39.174.50	16	10.39.0.1	12	Media7IF
9	Media	IPv4	10.40.174.50	16	10.40.0.1	13	Media8IF
10	Media & Control	IPv4	10.41.174.50	16	0.0.0.0	14	MediaCtrl9IF
11	Media	IPv4	10.42.174.50	16	0.0.0.0	15	Media10IF
12	Media	IPv4	10.43.174.50	16	10.43.0.1	16	Media11IF
13	Media	IPv4	10.44.174.50	16	0.0.0.0	17	Media12IF
14	Media& Control	IPv4	10.45.174.50	16	10.45.0.1	18	Media13IF

Complementing the network configuration are some VLAN-related parameters, determining if VLANs are enabled and the 'Native' VLAN ID (see the sub-sections below) as well as VLAN priorities and DiffServ values for the supported Classes Of Service.

Each row of the table defines a logical IP interface with its own IP address, subnet mask (represented by Prefix Length), VLAN ID, name, and application types that are allowed on this interface. Multiple interfaces can be defined with a default gateway. Traffic from this interface destined to a subnet which does not meet any of the routing rules (either local or static routes) are forwarded to this gateway.

Index Column:

This column holds the index of each interface. Possible values are 0 to 47. Each interface index must be unique.

Application Types Column:

This column defines the types of applications that are allowed on this interface:

OAMP – Operations, Administration, Maintenance and Provisioning applications such as Web, Telnet, SSH, SNMP

CONTROL – Call Control protocols (i.e., SIP)

MEDIA – RTP streams of voice

Various combinations of the above mentioned types

The following table shows the possible values of this column and their descriptions:

Table 11-3: Application Types

Value	Description
0	OAMP: only OAMP applications are allowed on this interface.
1	MEDIA: only Media (RTP) are allowed on this interface.
2	CONTROL: only Call Control applications are allowed on this interface.
3	OAMP & MEDIA: only OAMP and Media (RTP) applications are allowed on this interface.
4	OAMP & CONTROL: only OAMP and Call Control applications are allowed on this interface.
5	MEDIA & CONTROL: only Media (RTP) and Call Control applications are allowed on this interface.
6	OAMP, MEDIA & CONTROL: all of the application types are allowed on this interface.
99	MAINTENANCE: only HA maintenance applications are allowed this interface.

For valid configuration guidelines, see 'Multiple Interface Table Configuration Summary and Guidelines' on page 78.

Interface Mode Column:

The Interface Mode column determines the method that this interface uses to acquire its IP address. For IPv4 Manual IP Address assignment, use "IPv4 Manual" (10).

IP Address and Prefix Length Columns:

These columns allow the user to configure an IPv4 / IPv6 IP address and its related subnet mask.

The Prefix Length column holds the Classless Inter-Domain Routing (CIDR)-style representation of a dotted-decimal subnet notation. The CIDR-style representation uses a suffix indicating the number of bits which are set in the dotted-decimal format, in other words, 192.168.0.0/16 is synonymous with 192.168.0.0 and a subnet 255.255.0.0 (Refer to http://en.wikipedia.org/wiki/Classless_Inter-Domain_Routing for more information).

This CIDR notation lists the number of '1' bits in the subnet mask. So, a subnet mask of 255.0.0.0 (when broken down to its binary format) is represented by a prefix length of 8 (11111111 00000000 00000000 00000000), and a subnet mask of 255.255.255.252 is represented by a prefix length of 30 (11111111 11111111 11111111 11111100).

Each interface must have its own address space. Two interfaces may not share the same address space, or even part of it. The IP address should be configured as a dotted-decimal notation.

For IPv4 interfaces, the prefix length values range from 0 to 30.For IPv6 interfaces, the prefix length must be set to 64.

Gateway Column:

This column defines a default gateway for each interface. A default gateway can be defined for each interface. When traffic is sent from this interface to an unknown destination (i.e., not in the same subnet and not defined for any static routing rule), it is forwarded to this default gateway. The default gateway's address must be on the same subnet as the interface address. A separate routing table allows configuring additional static routing rules. See 'Configuring the IP Routing Table' on page 86for more details.

Note: In the example below, the default gateway for the OAMP application is 192.168.0.1, whereas for Media & Control applications it is 200.200.85.1.

Table 11-4: Configured Default Gateway Example

Index	Application Type	Interface Mode	IP Address	Prefix Length	Gateway	VLAN ID	Interface Name
0	OAMP	IPv4 Manual	192.168.0.2	16	192.168.0.1	100	Mgmt
1	Media & Control	IPv4 Manual	200.200.85.14	24	200.200.85.1	200	CntrlMedia

A separate routing table allows configuring static routing rules. Configuring the following routing enable OAMP applications to access peers on subnet 17.17.0.0 through the gateway 192.168.10.1 (which is not the default gateway of the interface), and Media & Conrol applications to access peers on subnet 171.79.39.0 through the gateway 200.200.85.10 (which is not the default gateway of the interface).

Table 11-5: Separate Routing Table Example

Destination	Prefix Length	Gateway	Interface	Metric	Status
17.17.0.0	16	192.168.10.1	0	1	Active
171.79.39.0	24	200.200.85.10	1	1	Active

VLAN ID Column:

This column defines the VLAN ID for each interface. This column must hold a unique value for each interface of the same address family.One IPv4 interface and one IPv6 interface may share the same VLAN ID, allowing hybrid networks on a single broadcast domain.

Interface Name Column:

This column allows the configuration of a short string (up to 16 characters) to name this interface. This name is displayed in management interfaces (Web, CLI, and SNMP) and is used in the Media Realm table. This column must have a unique value for each interface (no two interfaces can have the same name) and must not be left blank.

Primary / Secondary DNS Server IP Address Columns:

Defines the primary and secondary DNS server IP addresses for translating domain names into IP addresses.

11.2.1.1.2 Booting using DHCP

The *DHCPEnable* parameter enables the device to boot while acquiring an IP address from a DHCP server. Note that when using this method, Multiple Interface table/VLANs and other advanced configuration options are disabled.

11.2.1.1.3 Quality of Service Parameters

The device allows you to specify values for Layer-3 priorities, by assigning values to the following service classes:

Premium Media service class – used for RTP Media traffic

Premium Control Service class – used for Call Control traffic

Gold Service class – used for streaming applications

Bronze Service class – used for OAMP applications

The Layer-3 QoS parameters define the values of the DiffServ field in the IP Header of the frames related to a specific service class. The Layer-2 QoS parameters defines the values for the 3 priority bits in the VLAN tag (according to the IEEE 802.1p standard) according to the value of the DiffServ field found in the packet IP header.

The DiffServ Table (DiffServToVlanPriority) allows you to configure DiffServ-to-VLAN Priority mapping (Layer 2 class of service). For each packet sent to the LAN, the VLAN Priority of the packet is set according to the DiffServ value in the IP header of the packet. For Layer-3 CoS, you can use the PremiumServiceClassMediaDiffServ, PremiumServiceClassControlDiffServ, GoldServiceClassDiffServ, and BronzeServiceClassDiffServ parameters.

The mapping of an application to its CoS and traffic type is shown in the table below:

Table 11-6: Traffic/Network Types and Priority

Application	Traffic / Network Types	Class-of-Service (Priority)
Debugging interface	Management	Bronze
Telnet	Management	Bronze
DHCP	Management	Network
Web server (HTTP)	Management	Bronze
SNMP GET/SET	Management	Bronze
Web server (HTTPS)	Management	Bronze
RTP traffic	Media	Premium media
RTCP traffic	Media	Premium media
T.38 traffic	Media	Premium media
SIP	Control	Premium control
SIP over TLS (SIPS)	Control	Premium control
Syslog	Management	Bronze
SNMP Traps	Management	Bronze

Application	Traffic / Network Types	Class-of-Service (Priority)
DNS client	Varies according to DNS settings: OAMP Control	Depends on traffic type: Control: Premium Control Management: Bronze
NTP	Varies according to NTP settings (EnableNTPasOAM): OAMP Control	Depends on traffic type: Control: Premium control Management: Bronze

11.2.1.1.4 Assigning NTP Services to Application Types

NTP applications can be associated with different application types (OAMP or Control) in different setups. The table below describes the parameter for configuring this:

Table 11-7: Application Type Parameters

Parameter	Description
EnableNTPasOAM	Determines the application type for NTP services. **[1]** = OAMP (default) **[0]** = Control. **Note:** For this parameter to take effect, a device reset is required.

11.2.1.1.5 Multiple Interface Table Configuration Summary and Guidelines

Multiple Interface table configuration must adhere to the following rules:

Up to 48 different interfaces may be defined.

The indices used must be in the range between 0 and 47.

Each interface must have its own subnet. Defining two interfaces with addresses in the same subnet (i.e. two interfaces with 192.168.0.1/16 and 192.168.100.1/16) is illegal.

Subnets in different interfaces must not be overlapping in any way (i.e. defining two interfaces with 10.0.0.1/8 and 10.50.10.1/24 is invalid). Each interface must have its own address space.

The Prefix Length replaces the dotted decimal Subnet Mask presentation. This column must have a value of 0-30 for IPv4 interfaces.

Only one IPv4 interface with OAMP "Application Types" **must** be configured. At least one IPv4 interface with CONTROL "Application Types" **must** be configured. At least one IPv4 interface with MEDIA "Application Types" **must** be configured. These application types **may** be mixed (i.e. OAMP and CONTROL). Here are some examples for interface configuration:

- One IPv4 interface with "Application Types" OAMP, MEDIA & CONTROL (without VLANs).
- One IPv4 interface with "Application Types" OAMP, one other or more IPv4 interfaces with "Application Types" CONTROL, and one or more IPv4 interfaces with "Application Types" MEDIA (with VLANs).
- One IPv4 interface with "Application Types" OAMP & MEDIA, one other or more IPv4 interfaces with "Application Types" MEDIA & CONTROL.
- Other configurations are also possible while keeping to the above-mentioned rule.

Each network interface may be defined with a default gateway. This default gateway address must be in the same subnet as the associated interface. Additional routing rules may be specified in the Routing table ('Configuring the IP Routing Table' on page 86).

The Interface Name column may have up to 16 characters. This column allows the user to name each interface with an easier name to associate the interface with. This column must have a unique value to each interface and must not be left blank.

Primary and Secondary DNS server address may be configured for each interface. **Note:** Currently, the device supports DNS configuration for only one interface.

For IPv4 interfaces, the "Interface Mode" column must be set to "IPv4 Manual" (numeric value 10).

Quality of Service parameters specify the DiffServ field in the IP header according to service classes. DiffServ-to-VLAN Priority mapping allows associating each DiffServ value with a VLAN priority (according to IEEE 802.1p standard).

Network Configuration changes are offline. The new configuration should be saved and becomes available at the next startup.

Upon system start up, the Multiple Interface table is parsed and passes comprehensive validation tests. If any errors occur during this validation phase, the device sends an error message to the Syslog server and falls back to a "safe mode", using a single interface and no VLANs. Ensure that you view the Syslog messages that the device sends in system startup to see if any errors occurred.

Note: When configuring the device using the Web interface, it is possible to perform a quick validation of the configured Multiple Interface table and VLAN definitions, by clicking the **Done** button in the Multiple Interface Table Web page. It is highly recommended to perform this when configuring Multiple Interfaces and VLANs, using the Web Interface to ensure the configuration is complete and valid.

11.2.1.1.6 Troubleshooting the Multiple Interface Table

If any of the Multiple Interface table guidelines are violated, the device falls back to a "safe mode" configuration, working temporarily with IP address 192.168.0.2. For more information on validation failures, consult the Syslog messages.

Validation failures may be caused by one of the following:

One of the Application Types (OAMP, CONTROL, MEDIA) is missing in the IPv4 interfaces.

There are too many interfaces with "Application Types" of OAMP. Only one interface defined but the "Application Types" column is not set to "OAM + Media + Control" (numeric value 6).

An IPv4 interface was defined with "Interface Type" different than "IPv4 Manual" (10).

Two interfaces have the exact VLAN ID value.

Two interfaces have the same name.

Two interfaces share the same address space or subnet.

Apart from these validation errors, connectivity problems may be caused by one of the following:

Trying to access the device with untagged traffic when VLANs are on and Native VLAN is not configured properly.

Routing Table is not configured properly.

11.2.1.2 Setting Up VoIP Networking

11.2.1.2.1 Using the ini File

When configuring the network configuration using the *ini* File, use a textual presentation of the Interface and Routing Tables, as well as some other parameters. The following shows an example of a full network configuration, consisting of **all** the parameters described in this section:

```
;  *** TABLE InterfaceTable ***
[ InterfaceTable ]
FORMAT InterfaceTable_Index = InterfaceTable_ApplicationTypes,
InterfaceTable_InterfaceMode, InterfaceTable_IPAddress,
InterfaceTable_PrefixLength, InterfaceTable_Gateway, InterfaceTable_VlanID,
InterfaceTable_InterfaceName, InterfaceTable_PrimaryDNSServerIPAddress,
InterfaceTable_SecondaryDNSServerIPAddress,
InterfaceTable_UnderlyingInterface;
InterfaceTable 0 = 0, 10, 192.168.0.2, 16, 192.168.0.1, 501, Mgmt, , , ;
InterfaceTable 1 = 5, 10, 10.32.174.50, 24, 10.32.174.1, 2012, MediaCntrl, ,
, ;
InterfaceTable 2 = 1, 10, 10.33.174.50, 24, 10.33.174.1, 2013, Media1, , , ;
InterfaceTable 3 = 2, 10, 10.34.174.50, 24, 10.34.174.1, 2014, Cntrl1, , , ;
[ \InterfaceTable ]

; Routing Table Configuration:
[ StaticRouteTable ]
FORMAT StaticRouteTable_Index = StaticRouteTable_InterfaceName,
StaticRouteTable_Destination, StaticRouteTable_PrefixLength,
StaticRouteTable_Gateway, StaticRouteTable_Description;
StaticRouteTable 0 = 0, 10.31.174.0, 24, 192.168.11.1, ;
StaticRouteTable 1 = 1, 174.96.151.15, 24, 10.32.174.12, ;
StaticRouteTable 2 = 3, 10.35.174.0, 24, 10.34.174.240, ;
[ \StaticRouteTable ]

; Layer 3 QoS parameters (DiffServ):
PremiumServiceClassMediaDiffServ = 46
PremiumServiceClassControlDiffServ = 40
GoldServiceClassDiffServ = 26
BronzeServiceClassDiffServ = 10

;  *** TABLE DiffServToVlanPriority ***
[ DiffServToVlanPriority ]
FORMAT DiffServToVlanPriority_Index = DiffServToVlanPriority_DiffServ,
DiffServToVlanPriority_VlanPriority;
DiffServToVlanPriority 0 = 0, 7;
DiffServToVlanPriority 1 = 46, 6;
DiffServToVlanPriority 2 = 40, 6;
DiffServToVlanPriority 3 = 26, 4;
DiffServToVlanPriority 4 = 10, 2;
[ \DiffServToVlanPriority ]

; Application Type for NTP applications:
EnableNTPasOAM = 1
```

This *ini* file shows the following:

A Multiple Interface table with a an interface for OAMP applications (192.168.0.2/16), an

interface for Media & Control applications (10.32.174.50/24), an interface for Media applications (10.33.174.50/24), and an interface for Control applications (10.34.174.50/24).

Each interface is defined with its own VLAN ID, Default Gateway, and name.

A Routing table is configured with three static routing rules:

- Directing all traffic destined to subnet 10.31.174.0/24 to 192.168.11.1 over Interface Index 0
- Directing all traffic destined to subnet 174.96.151.15/24 to 10.32.174.12 over Interface Index 1
- Directing all traffic destined to subnet 10.35.174.0/24 to 10.34.174.240 over Interface Index 3

Layer-3 QoS values are assigned:

- For Media Service class, the default DiffServ value is set to 46
- For Control Service class, the default DiffServ value is set to 40
- For Gold Service class, the default DiffServ value is set to 26
- For Bronze Service class, the default DiffServ value is set to 10

Layer-2 QoS values are assigned:

- For packets sent with DiffServ value of 46, set VLAN priority to 6
- For packets sent with DiffServ value of 40, set VLAN priority to 6
- For packets sent with DiffServ value of 26, set VLAN priority to 4
- For packets sent with DiffServ value of 10, set VLAN priority to 2

The NTP applications are configured to serve as OAMP applications.

Notes:

Lines that begin with a semicolon are considered a remark and are ignored.

When using the *ini* file, the Multiple Interface table and the DiffServ To VLAN Priority table must have the prefix and suffix to allow the INI File parser to correctly recognize and parse the table.

11.2.1.2.2 Networking Configuration Examples

This section provides examples of network configurations (and their corresponding *ini* file configuration).

Example 1 - One VoIP Interface for All Applications: Multiple Interface table with a single interface for OAMP, Media and Control applications:

Table 11-8: Multiple Interface Table - Example 1

Index	Allowed Applications	Interface Mode	IP Address	Prefix Length	Default Gateway	VLAN ID	Interface Name
0	OAMP, Media & Control	IPv4	192.168.0.2	16	192.168.0.1	1	myInterface

Additional static routing rules:

Table 11-9: Routing Table - Example 1

Destination	Prefix Length	Gateway	Interface	Metric
201.201.0.0	16	192.168.11.10	0	1
202.202.0.0	16	192.168.11.1	0	1

The NTP applications remain with their default application types.

The corresponding *ini* file configuration is shown below:

```
; Interface Table Configuration:
[InterfaceTable]
FORMAT InterfaceTable_Index = InterfaceTable_ApplicationTypes,
InterfaceTable_InterfaceMode, InterfaceTable_IPAddress,
InterfaceTable_PrefixLength, InterfaceTable_Gateway, InterfaceTable_VlanID,
InterfaceTable_InterfaceName, InterfaceTable_PrimaryDNSServerIPAddress,
InterfaceTable_SecondaryDNSServerIPAddress, InterfaceTable_UnderlyingInterface;
InterfaceTable 0  = 6, 10, 192.168.0.2, 16, 192.168.0.1, 1, myInterface, , ,
;
[\InterfaceTable]

; Routing Table Configuration:
[ StaticRouteTable ]
FORMAT StaticRouteTable_Index = StaticRouteTable_InterfaceName,
StaticRouteTable_Destination, StaticRouteTable_PrefixLength,
StaticRouteTable_Gateway, StaticRouteTable_Description;
StaticRouteTable 0 = 0, 201.201.0.0, 16, 192.168.11.10, ;
StaticRouteTable 1 = 0, 202.202.0.0, 16, 192.168.11.1, ;
[ \StaticRouteTable ]
```

Example 2 - Three VoIP Interfaces, One for each Application Exclusively: the Multiple Interface table is configured with three interfaces, one exclusively for each application type: one interface for OAMP applications, one for Call Control applications, and one for RTP Media applications:

Table 11-10: Multiple Interface Table - Example 2

Index	Allowed Applications	Interface Mode	IP Address	Prefix Length	Default Gateway	VLAN ID	Interface Name
0	OAMP	IPv4 Manual	192.168.0.2	16	192.168.0.1	1	ManagementIF
1	Control	IPv4 Manual	200.200.85.14	24	200.200.85.1	200	myControlIF
2	Media	IPv4 Manual	211.211.85.14	24	211.211.85.1	211	myMediaIF

Additional static routing rules:

Table 11-11: Routing Table - Example2

Destination	Prefix Length	Gateway	Interface	Metric
176.85.49.0	24	192.168.11.1	0	1

All other parameters are set to their respective default values. The NTP application remains with its default application types.

The corresponding *ini* file configuration is shown below:

```
; Interface Table Configuration:
[InterfaceTable]
FORMAT InterfaceTable_Index = InterfaceTable_ApplicationTypes,
InterfaceTable_InterfaceMode, InterfaceTable_IPAddress,
InterfaceTable_PrefixLength, InterfaceTable_Gateway, InterfaceTable_VlanID,
InterfaceTable_InterfaceName, InterfaceTable_PrimaryDNSServerIPAddress,
InterfaceTable_SecondaryDNSServerIPAddress, InterfaceTable_UnderlyingInterface;
InterfaceTable 0  = 0, 10, 192.168.0.2,  16, 192.168.0.1, 1, ManagementIF, , ,;
InterfaceTable 1  = 2, 10, 200.200.85.14,  24, 200.200.85.1, 200,
myControlIF, , ,;
InterfaceTable 2  = 1, 10, 211.211.85.14,  24, 211.211.85.1, 211, myMediaIF,
, ,;
[\InterfaceTable]
; Routing Table Configuration:
[ StaticRouteTable ]
FORMAT StaticRouteTable_Index = StaticRouteTable_InterfaceName,
StaticRouteTable_Destination, StaticRouteTable_PrefixLength,
StaticRouteTable_Gateway, StaticRouteTable_Description;
StaticRouteTable 0 = 0, 176.85.49.0, 24, 192.168.11.1, ;
[ \StaticRouteTable ]
```

Example 3 - Three Interfaces: one exclusively for management (OAMP applications) and two others for Call Control and RTP (Control and Media applications) :

Table 11-12: Multiple Interface Table - Example 3

Index	Allowed Applications	Interface Mode	IP Address	Prefix Length	Default Gateway	VLAN ID	Interface Name
0	OAMP	IPv4 Manual	192.168.0.2	16	192.168.0.1	1	Mgmt

Index	Allowed Applications	Interface Mode	IP Address	Prefix Length	Default Gateway	VLAN ID	Interface Name
1	Media & Control	IPv4 Manual	200.200.85.14	24	200.200.85.1	201	MediaCntrl1
2	Media & Control	IPv4 Manual	200.200.86.14	24	200.200.86.1	202	MediaCntrl2

Additional static routing rules

Table 11-13: Routing Table - Example 3

Destination	Destination Subnet Mask/Prefix Length	Gateway	Interface	Metric
176.85.49.0	24	192.168.0.10	0	1

All other parameters are set to their respective default values. The NTP application remains with its default application types.

The corresponding *ini* file configuration is shown below:

```
; Interface Table Configuration:
[InterfaceTable]
FORMAT InterfaceTable_Index = InterfaceTable_ApplicationTypes,
InterfaceTable_InterfaceMode, InterfaceTable_IPAddress,
InterfaceTable_PrefixLength, InterfaceTable_Gateway, InterfaceTable_VlanID,
InterfaceTable_InterfaceName, InterfaceTable_PrimaryDNSServerIPAddress,
InterfaceTable_SecondaryDNSServerIPAddress, InterfaceTable_UnderlyingInterface;
InterfaceTable 0  = 0, 10, 192.168.0.2,  16, 192.168.0.1, 1, Mgmt,,,;
InterfaceTable 1  = 5, 10, 200.200.85.14,  24, 200.200.85.1, 201,
MediaCntrl1,,,;
InterfaceTable 2  = 5, 10, 200.200.86.14,  24, 200.200.86.1, 202,
MediaCntrl2,,,;

[\InterfaceTable]
; Routing Table Configuration:
[ StaticRouteTable ]
FORMAT StaticRouteTable_Index = StaticRouteTable_InterfaceName,
StaticRouteTable_Destination, StaticRouteTable_PrefixLength,
StaticRouteTable_Gateway, StaticRouteTable_Description;
StaticRouteTable 0 = 0, 176.85.49.0, 24, 192.168.11.1, ;
[ \StaticRouteTable ]
```

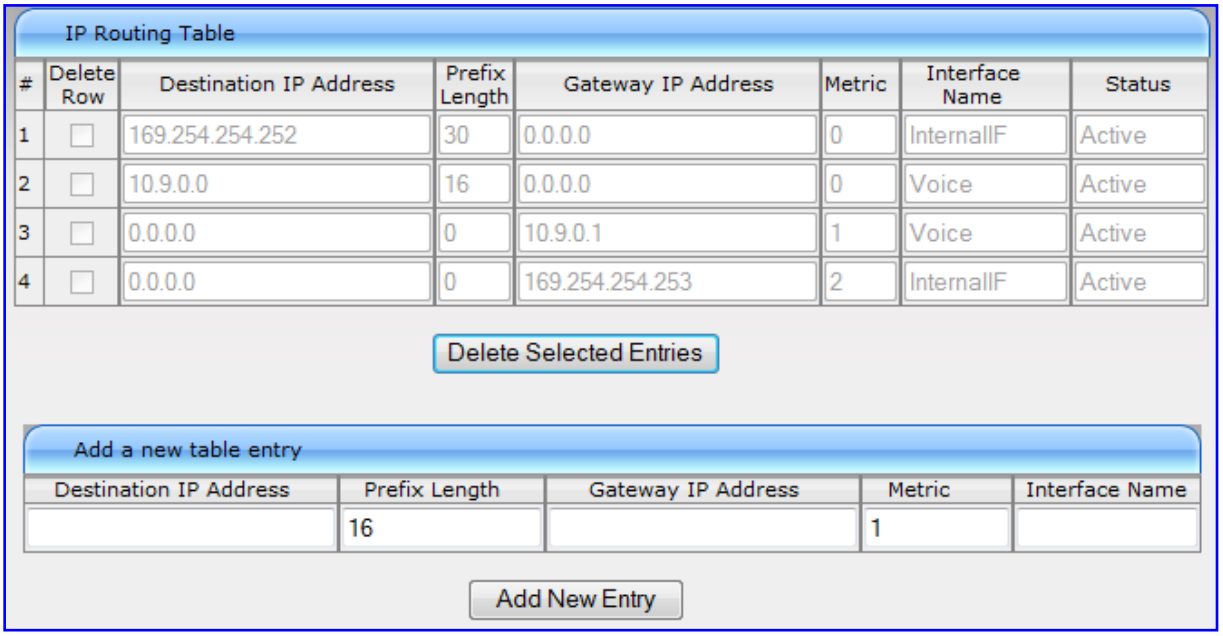
11.3 Configuring the IP Routing Table

The IP Routing Table page allows you to define up to 30 static IP routing rules for the device. These rules can be associated with a network interface (defined in the Multiple Interface table) and therefore, the routing decision is based on the source subnet/VLAN. If not associated with an IP interface, the static IP rule is based on destination IP address.

To configure static IP routing:

1. Open the IP Routing Table page (**Configuration** tab >**VoIP** menu >**Network** submenu >**IP Routing Table**).

Figure 11-4: IP Routing Table Page


IP Routing Table
| Delete Row | Destination IP Address | Prefix Length | Gateway IP Address | Metric | Interface Name | Status
1 | 169.254.254.252 | 30 | 0.0.0.0 | 0 | InternalIF | Active
2 | 10.9.0.0 | 16 | 0.0.0.0 | 0 | Voice | Active
3 | 0.0.0.0 | 0 | 10.9.0.1 | 1 | Voice | Active
4 | 0.0.0.0 | 0 | 169.254.254.253 | 2 | InternalIF | Active
Delete Selected Entries
Add a new table entry
Destination IP Address | Prefix Length | Gateway IP Address | Metric | Interface Name
16 | 1
Add New Entry

In the Add a new table entry table, add a new static routing rule according to the parameters described in the table below.

Click **Add New Entry**; the new routing rule is added to the IP routing table.

To delete a routing rule from the table, select the 'Delete Row' check box corresponding to the required routing rule, and then click **Delete Selected Entries**.

Notes:

You can delete only inactive routing rules.

You can also configure the IP Routing table using the *ini* file table parameter StaticRouteTable.

Table 11-14: IP Routing Table Description

Parameter	Description
Destination IP Address **[StaticRouteTable_Destination]**	Specifies the IP address of the destination host/network.
Prefix Length **[StaticRouteTable_PrefixLength]**	Specifies the subnet mask of the destination host/network.

Parameter	Description
The address of the host/network you want to reach is determined by an AND operation that is applied to the fields 'Destination IP Address' and 'Destination Mask'. For example, to reach the network 10.8.x.x, enter 10.8.0.0 in the field 'Destination IP Address' and 255.255.0.0 in the field 'Destination Mask'. As a result of the AND operation, the value of the last two octets in the field 'Destination IP Address' is ignored. To reach a specific host, enter its IP address in the field 'Destination IP Address' and 255.255.255.255 in the field 'Destination Mask'.	
Gateway IP Address **[StaticRouteTable_Gateway]**	The IP address of the router (next hop) to which the packets are sent if their destination matches the rules in the adjacent columns. **Note:** The Gateway address must be in the same subnet as the IP address of the interface over which you configure this static routing rule.
Metric	The number of hops needed to get to the specified destination. **Note:** The recommended value for this parameter is 1.
Interface Name **[StaticRouteTable_InterfaceName]**	Assigns a network interface to this routing rule. This value is the index of the network interface as defined in the Multiple Interface table (see 'Configuring IP Interface Settings' on page 68). **Note:** The IP address of the 'Gateway IP Address' field must be in the same subnet as this interface's IP address.
Status	Read-only field displaying the status of the static IP route: "Active" - routing rule is used ny the device "Inactive" - routing rule is not applied

11.3.1 Routing Table Columns

Each row of the Routing table defines a static routing rule. Traffic destined to the subnet specified in the routing rule is re-directed to the defined gateway, reachable through the specified interface.

The IP Routing table consists of the following:

Table 11-15: IP Routing Table Layout

Destination	Prefix Length	Gateway	Interface	Metric	Status
201.201.0.0	16	192.168.0.1	0	1	Active
202.202.0.0	16	192.168.0.2	0	1	Active
203.203.0.0	16	192.168.0.3	0	1	Active
225.225.0.0	16	192.168.0.25	0	1	Inactive

11.3.1.1 Destination Column

This column defines the destination of the route rule. The destination can be a single host or a whole subnet, depending on the Prefix Length/Subnet Mask specified for this routing rule.

11.3.1.2 Prefix Length Column

The Prefix Length column holds the Classless Inter-Domain Routing (CIDR)-style representation of a dotted-decimal subnet notation. The CIDR-style representation uses a suffix indicating the number of bits that are set in the dotted-decimal format. For example, 16 is synonymous with subnet 255.255.0.0.

11.3.1.3 Gateway Column

The Gateway column defines the IP address of the next hop used for traffic destined to the subnet/host as defined in the destination/mask columns. This gateway address must be on the same subnet as the IP address of the interface configured in the Interface column.

11.3.1.4 Interface Column

This column defines the interface index (in the Multiple Interface table) from which the gateway address is reached.

Note: The Interface Address family must be coherent with the Routing Rule Address family. IPv4 interfaces cannot be selected in an IPv6 routing rule, and vice versa.

The VoIP Interface Table:

Index	Allowed Application Types	Interface Mode	IP Address	Prefix Length	Gateway	VLAN ID	Interface Name
0	OAMP	IPv4 Manual	192.168.0.2	16	192.168.0.1	501	Mgmt
1	Media & Control	IPv4 Manual	10.32.174.50	24	10.32.174.1	2012	MediaCntrl
2	Media	IPv4 Manual	10.33.174.50	24	10.33.174.1	2013	Media1
3	Control	IPv4 Manual	10.34.174.50	24	10.34.174.1	2014	Cntrl1

The VoIP Static Routing Table:

Destination	Destination Subnet Mask / Prefix Length	Gateway (Next Hop) IP Address	Metric	Interface Index
10.31.174.0	24	192.168.11.1	1	0
174.96.151.15	24	10.32.174.12	1	1
10.35.174.0	24	10.34.174.240	1	3

The IP Address of the Gateway (Next Hop) must reside at the same subnet as the IP Address of the interface configured in the Interface Index column

11.3.1.5 Metric Column

The Metric column must be set to 1 for each static routing rule.

11.3.1.6 State Column

The State column displays the state of each static route. Possible values are "Active" and "Inactive". When the destination IP address is not on the same segment with the next hop or the interface does not exist, the route state changes to "Inactive".

11.3.2 Routing Table Configuration Summary and Guidelines

The Routing table configurations must adhere to the following rules:

Up to 30 different static routing rules may be defined.

The Prefix Length replaces the dotted-decimal subnet mask presentation. This column must have a value of 0-31 for IPv4 interfaces and a value of 64 for IPv6 interfaces.

The "Gateway" IP Address must be on the same subnet as the IP address of the interfaces configured in the Interface Index column.

The “Metric” column must be set to 1.

Network Configuration changes are offline. The new configuration should be saved and will be available at the next startup.

11.3.3 Troubleshooting the Routing Table

When adding a new static routing rule, the added rule passes a validation test. If errors are found, the routing rule is rejected and is not added to the IP Routing table. Failed routing validations may result in limited connectivity (or no connectivity) to the destinations specified in the incorrect routing rule. For any error found in the Routing table or failure to configure a routing rule, the device sends a notification message to the Syslog server reporting the problem.

Common routing rule configuration errors may include the following:

The IP address specified in the "Gateway" column is unreachable from the interface specified in the "Interface" column.

The same destination is defined in two different routing rules.

More than 30 routing rules were defined.

Note: If a routing rule is required to access OAMP applications (for remote management, for instance) and this route is not configured correctly, the route is not added and the device is not accessible remotely. To restore connectivity, the device must be accessed locally from the OAMP subnet and the required routes be configured.

11.4 Configuring QoS Settings

The Diff Serv Table page is used for configuring the Layer-2 and Layer-3 Quality of Service (QoS) parameters for VoIP. DiffServ is an architecture providing different types or levels of service for IP traffic. DiffServ (according to RFC 2474), prioritizes certain traffic types based on their priority, thereby, accomplishing a higher-level QoS at the expense of other traffic types. By prioritizing packets, DiffServ routers can minimize transmission delays for time-sensitive packets such as VoIP packets.

This page allows you to assign Differentiated Services (DiffServ) to four classes of traffic (Media Premium, Control Premium, Gold, and Bronze) and to assign VLAN priorities (IEEE 802.1p) to various values of DiffServ. For a detailed description of the parameters appearing on this page, see 'Networking Parameters' on page 280. For a description on QoS and the mapping of each application to a class of service, see 'Quality of Service Parameters' on page 77.

Notes:

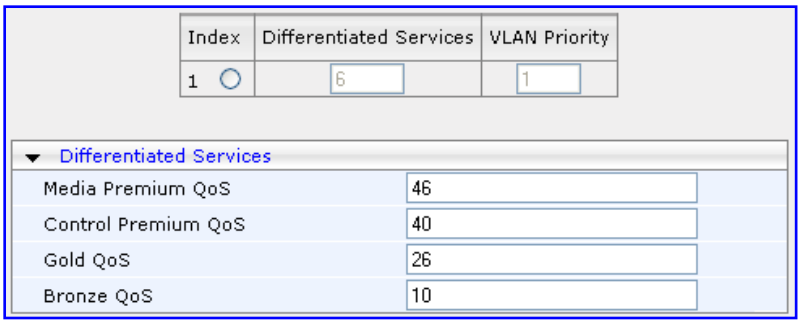
For the settings of this table to take effect, a device reset is required.

You can also configure the DiffServ table using the ini file table parameter DiffServToVlanPriority.

To configure QoS:

1. Open the Diff Serv Table page (**Configuration** tab >**VoIP** menu >**Network** submenu >**QoS Settings**).

Figure 11-5: DiffServ Table Page



Configure DiffServ to VLAN priority mapping (Layer-2 QoS):

a. Enter an index entry, and then click Add.

In the 'Differentiated Services' field, enter the DiffServ value (0-63) and its corresponding VLAN priority level (0-7).

Click Apply.

Configure the desired DiffServ (Layer-3 QoS) values for the following traffic classes:

Media Premium QoS: this affects Media RTP packets sent by the VoIP towards the LAN.

Control Premium QoS: this affects Control Protocol (SIP) packets sent by the VoIP towards the LAN.

Gold QoS: this affects HTTP Streaming packets sent by the VoIP towards the LAN.

Bronze QoS: this affects OAMP packets sent by the VoIP towards the LAN.

Click Submit to apply your changes.

Save the changes to flash memory and reset the device (see 'Saving Configuration' on page 245).

11.5 DNS

You can use the device's embedded domain name server (DNS) or an external, third-party DNS to translate domain names into IP addresses. This is useful if domain names are used as the destination in call routing.

The device supports the configuration of the following DNS types:

Internal DNS table - see 'Configuring the Internal DNS Table' on page 91

Internal SRV table - see 'Configuring the Internal SRV Table' on page 92

11.5.1 Configuring the Internal DNS Table

The Internal DNS Table page, similar to a DNS resolution translates up to 20 host (domain) names into IP addresses. Up to four different IP addresses can be assigned to the same host name (typically used for alternative call routing).

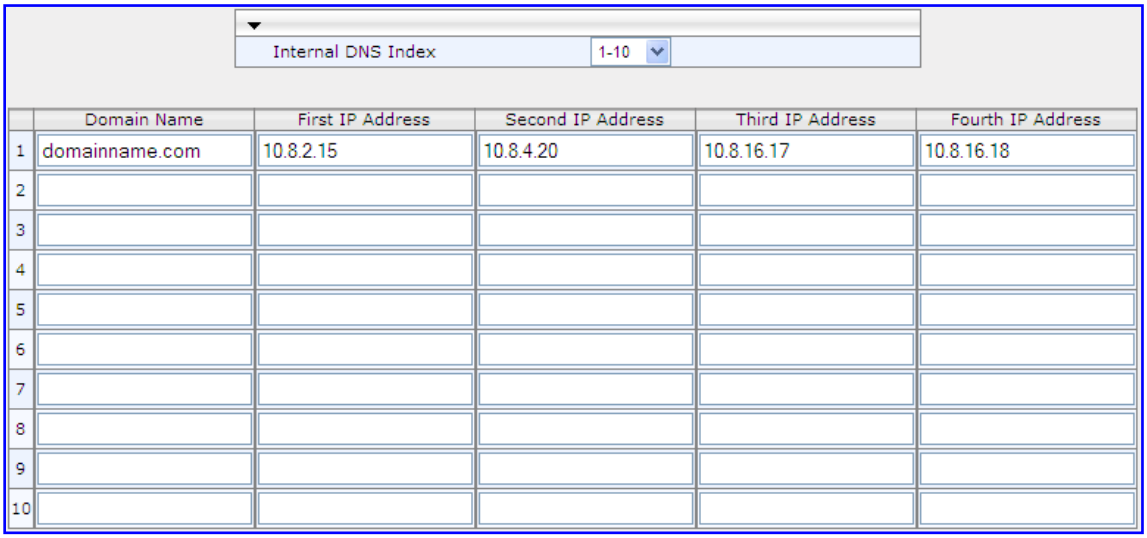
Notes:

- The device initially attempts to resolve a domain name using the Internal DNS table. If the domain name isn't listed in the table, the device performs a DNS resolution using an external DNS server (defined in the Multiple Interface table - see 'Configuring IP Interface Settings' on page 68).
- You can also configure the DNS table using the *ini* file table parameter DNS2IP (see 'DNS Parameters' on page 287).

To configure the internal DNS table:

1. Open the Internal DNS Table page (**Configuration** tab >**VoIP** menu >**Network** submenu >**DNS** submenu >**Internal DNS Table**).

Figure 11-6: Internal DNS Table Page



In the 'Domain Name' field, enter the host name to be translated. You can enter a string of up to 31 characters.

In the 'First IP Address' field, enter the first IP address (in dotted-decimal format notation) to which the host name is translated.

Optionally, in the 'Second IP Address', 'Third IP Address', and 'Second IP Address' fields, enter the next IP addresses to which the host name is translated.

Click Subm**it to** apply your changes.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

11.5.2 Configuring the Internal SRV Table

The Internal SRV Table page resolves host names to DNS A-Records. Three different A-Records can be assigned to each host name. Each A-Record contains the host name, priority, weight, and port.

Notes:

If the Internal SRV table is configured, the device initially attempts to resolve a domain name using this table. If the domain name isn't found, the device performs an Service Record (SRV) resolution using an external DNS server (defined in the Multiple Interface table - see 'Configuring IP Interface Settings' on page 68).

You can also configure the Internal SRV table using the *ini* file table parameter SRV2IP (see 'DNS Parameters' on page 287).

To configure the Internal SRV table:

1. Open the Internal SRV Table page (**Configuration** tab >**VoIP** menu >**Network** submenu >**DNS** submenu >**Internal SRV Table**).

Figure 11-7: Internal SRV Table Page

	Domain Name	Transport Type	DNS Name 1	Priority	Weight	Port	DNS Name 2	Priority	Weight	Port	DNS Name 3	Priority	Weight	Port
1		UDP												
2		UDP												
3		UDP												
4		UDP												
5		UDP												
6		UDP												
7		UDP												
8		UDP												
9		UDP												
10		UDP												

In the 'Domain Name' field, enter the host name to be translated. You can enter a string of up to 31 characters.

From the 'Transport Type' drop-down list, select a transport type.

In the 'DNS Name 1' field, enter the first DNS A-Record to which the host name is translated.

In the 'Priority', 'Weight' and 'Port' fields, enter the relevant values

Repeat steps 4 through 5, for the second and third DNS names, if required.

Repeat steps 2 through 6, for each entry.

Click Subm**it to** apply your changes.

To save the changes so they are available after a hardware reset or power fail, see 'Saving Configuration' on page 245.

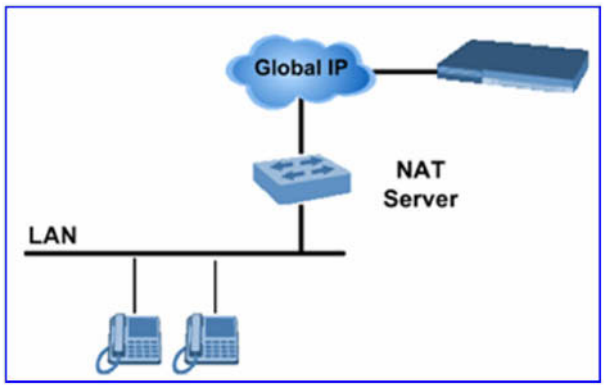
11.6 NAT (Network Address Translation) Support

Network Address Translation (NAT) is a mechanism that maps a set of internal IP addresses used within a private network to global IP addresses, providing transparent routing to end hosts. The primary advantages of NAT include (1) Reduction in the number of global IP addresses

required in a private network (global IP addresses are only used to connect to the Internet); (2) Better network security by hiding its internal architecture.

The following figure illustrates the device's supported NAT architecture.

Figure 11-8: NAT Support



The design of SIP creates a problem for VoIP traffic to pass through NAT. SIP uses IP addresses and port numbers in its message body and the NAT server can't modify SIP messages and therefore, can't change local to global addresses. Two different streams traverse through NAT: signaling and media. A device (located behind a NAT) that initiates a signaling path has problems in receiving incoming signaling responses (they are blocked by the NAT server). Furthermore, the initiating device must notify the receiving device where to send the media.

To resolve these issues, the following mechanisms are available:

First Incoming Packet Mechanism (see 'First Incoming Packet Mechanism' on page 93)

RTP No-Op packets according to the avt-rtp-noop draft (see 'No-Op Packets' on page 93)

For information on SNMP NAT traversal, refer to the *Product Reference Manual*.

11.6.1 First Incoming Packet Mechanism

If the remote device resides behind a NAT device, it's possible that the device can activate the RTP/RTCP/T.38 streams to an invalid IP address / UDP port. To avoid such cases, the device automatically compares the source address of the incoming RTP/RTCP/T.38 stream with the IP address and UDP port of the remote device. If the two are not identical, the transmitter modifies the sending address to correspond with the address of the incoming stream. The RTP, RTCP and T.38 can thus have independent destination IP addresses and UDP ports.

You can disable the NAT mechanism by setting the *ini* file parameter DisableNAT to 1. The two parameters EnableIpAddrTranslation and EnableUdpPortTranslation allow you to specify the type of compare operation that occurs on the first incoming packet. To compare only the IP address, set EnableIpAddrTranslation to 1, and EnableUdpPortTranslation to 0. In this case, if the first incoming packet arrives with only a difference in the UDP port, the sending addresses won't change. If both the IP address and UDP port need to be compared, then both parameters need to be set to 1.

11.6.2 No-Op Packets

The device's No-Op packet support can be used to verify Real-Time Transport Protocol (RTP) and T.38 connectivity, and to keep NAT bindings and Firewall pinholes open. The No-Op packets are available for sending in RTP and T.38 formats.

You can control the activation of No-Op packets by using the *ini* file parameter NoOpEnable. If No-Op packet transmission is activated, you can control the time interval in which No-Op packets are sent in the case of silence (i.e., no RTP or T.38 traffic). This is performed using the *ini* file parameter NoOpInterval. For a description of the RTP No-Op *ini* file parameters, see 'Networking Parameters' on page 280.

RTP No-Op: The RTP No-Op support complies with IETF Internet-Draft draft-wing-avt-rtp-noop-03 ("A No-Op Payload Format for RTP"). This IETF document defines a No-Op payload format for RTP. The draft defines the RTP payload type as dynamic. You can control the payload type with which the No-Op packets are sent. This is performed using the RTPNoOpPayloadType *ini* parameter (see 'Networking Parameters' on page 280). Alcatel-Lucent' default payload type is 120.

T.38 No-Op: T.38 No-Op packets are sent only while a T.38 session is activated. Sent packets are a duplication of the previously sent frame (including duplication of the sequence number).

Note: Receipt of No-Op packets is always supported.

11.7 Configuring NFS Settings

Network File System (NFS) enables the device to access a remote server's shared files and directories, and to handle them as if they're located locally. You can configure up to 16 different NFS file systems. As a file system, the NFS is independent of machine types, operating systems, and network architectures. NFS is used by the device to load the *cmp*, *ini*, and auxiliary files, using the Automatic Update mechanism (refer to the *Product Reference Manual*). Note that an NFS file server can share multiple file systems. There must be a separate row for each remote file system shared by the NFS file server that needs to be accessed by the device.

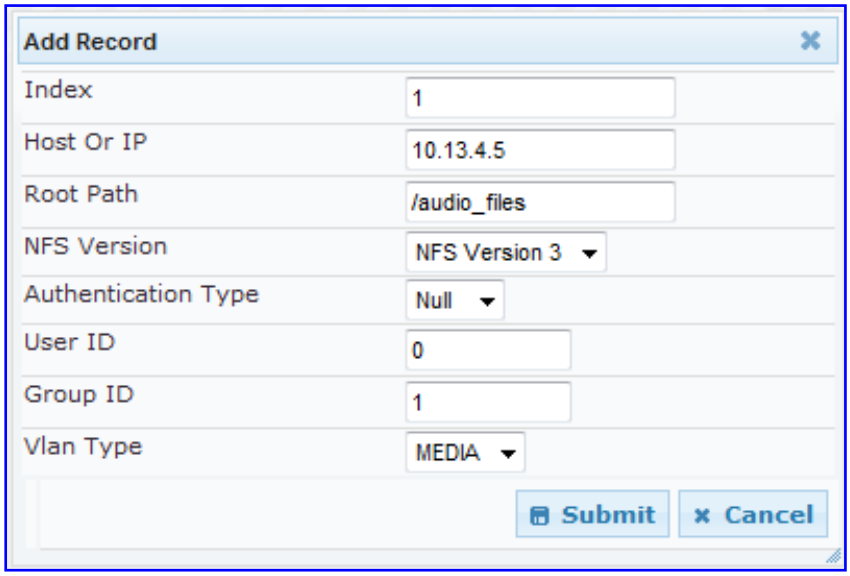
To add remote NFS file systems:

1. Open the Application Settings page (**Configuration** tab >**System** menu >**Application Settings**).

Under the NFS Settings group, click the **NFS Table** button; the NFS Settings page appears.

Click the **Add** button; the Add Record dialog box appears:

Figure 11-9: Add Record Dialog Box for NFS



Configure the NFS parameters according to the table below.

Click the **Submit** button; the remote NFS file system is immediately applied, which can be verified by the appearance of the 'NFS mount was successful' message in the Syslog server.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

Notes:

To avoid terminating current calls, a row must not be deleted or modified while the device is currently accessing files on that remote NFS file system.

The combination of 'Host Or IP' and 'Root Path' must be unique for each row in the table. For example, the table must include only one row with a Host/IP of 192.168.1.1 and Root Path of /audio.

For configuring Web interface tables, see 'Working with Tables' on page 28.

You can also configure the NFS table using the *ini* file table parameter NFSServers (see 'NFS Parameters' on page 286).

Table 11-16: NFS Settings Parameters

Parameter	Description
Index	The row index of the remote file system. The valid range is 1 to 16.
Host Or IP	The domain name or IP address of the NFS server. If a domain name is provided, a DNS server must be configured.
Root Path	Path to the root of the remote file system in the format: /**[path].** For example, '/audio'.
NFS Version	NFS version used to access the remote file system. **[2]** NFS Version 2 **[3]** NFS Version 3 (default)
Authentication Type	Authentication method used for accessing the remote file system. **[0]** Null **[1]** Unix (default)
User ID	User ID used in authentication when using Unix. The valid range is 0 to 65537. The default is 0.
Group ID	Group ID used in authentication when using Unix. The valid range is 0 to 65537. The default is 1.
VLAN Type	The VLAN type for accessing the remote file system. **[0]** OAM **[1]** MEDIA (default) **Note:** This parameter applies only if VLANs are enabled or if Multiple IPs is configured (see 'Network Configuration' on page 73).

11.8 Robust Receipt of Media Streams

This mechanism filters out unwanted RTP streams that are sent to the same port number on the device. These multiple RTP streams can result from traces of previous calls, call control errors, and deliberate attacks. When more than one RTP stream reaches the device on the same port number, the device accepts only one of the RTP streams and rejects the rest of the streams.

The RTP stream is selected according to the following: The first packet arriving on a newly opened channel sets the source IP address and UDP port from which further packets are received. Thus, the source IP address and UDP port identify the currently accepted stream. If a new packet arrives whose source IP address or UDP port are different to the currently accepted RTP stream, one of the following occurs:

The device reverts to the new RTP stream when the new packet has a source IP address and UDP port that are the same as the remote IP address and UDP port that were stated during the opening of the channel.

The packet is dropped when the new packet has any other source IP address and UDP port.

11.9 Multiple Routers Support

Multiple routers support is designed to assist the device when it operates in a multiple routers network. The device learns the network topology by responding to Internet Control Message Protocol (ICMP) redirections and caches them as routing rules (with expiration time).

When a set of routers operating within the same subnet serve as devices to that network and intercommunicate using a dynamic routing protocol, the routers can determine the shortest path to a certain destination and signal the remote host the existence of the better route. Using multiple router support, the device can utilize these router messages to change its next hop and establish the best path.

Note: Multiple Routers support is an integral feature that doesn't require configuration.

12 Security

This section describes the VoIP security-related configuration.

12.1 Configuring Firewall Settings

The device provides an internal firewall, allowing you (the security administrator) to define network traffic filtering rules. You can add up to 50 ordered firewall rules.

The access list provides the following firewall rules:

Block traffic from known malicious sources

Only allow traffic from known friendly sources, and block all others

Mix allowed and blocked network sources

Limit traffic to a pre-defined rate (blocking the excess)

Limit traffic to specific protocols, and specific port ranges on the device

For each packet received on the network interface, the table is scanned from the top down until a matching rule is found. This rule can either deny (*block*) or permit (*allow*) the packet. Once a rule in the table is located, subsequent rules further down the table are ignored. If the end of the table is reached without a match, the packet is accepted. For more information on the internal firewall, refer to the *Product Reference Manual*.

Notes:

It is recommended to add a rule at the end of your table that blocks all traffic and add firewall rules above it (in the table) that allow traffic (with bandwidth limitations). To block all traffic, the following must be set:
- IP address to 0.0.0.0
- Prefix length of 0 (implies the rule can match any IP address)
- Local port range 0-65535
- Protocol "Any"
- Action Upon Match "block"

If you are using the High Availability feature and you have added block rules, then ensure that you also add allow rules concerned with HA traffic. For more information, see 'Configuring Firewall Allowed Rules' on page 237.

You can also configure the firewall settings using the *ini* file table parameter AccessList (see 'Security Parameters' on page 302).

To add firewall rules:

1. Open the Firewall Settings page (**Configuration** tab >**VoIP** menu >**Security** submenu >**Firewall Settings**).

Figure 12-1: Firewall Settings Page

Edit Rule	Rule Status	Source IP	Source Port	Prefix Length	Local Port Range	Protocol	Use Specific Interface	Interface Name	Packet Size	Byte rate	Burst Bytes	Action Upon Match	Match Count
1	Not Active	mgmt.customer.com	0	32	0 - 80	tcp	Enable	Mng	0	0	0	Allow	0
2	Not Active	192.0.0.0	0	9	0-65535	Any	Disable	None	0	40000	50000	ALLOW	0
3	Not Active	10.31.4.0	0	24	4000-9000	Any	Disable	None	0	0	0	BLOCK	0
4	Not Active	10.4.0.0	0	16	4000-9000	Any	Disable	None	0	0	0	BLOCK	0

In the 'Add' field, enter the index of the access rule that you want to add, and then click **Add**; a new firewall rule index appears in the table.

Configure the firewall rule's parameters according to the table below.

Click one of the following buttons:

Apply: saves the new rule (without activating it).

Duplicate Rule: adds a new rule by copying a selected rule.

Activate: saves the new rule and activates it.

Delete: deletes the selected rule.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

The previous figure shows the following access list settings:

Rule #1: traffic from the host 'mgmt.customer.com' destined to TCP ports 0 to 80, is always allowed.

Rule #2: traffic from the 192.xxx.yyy.zzz subnet, is limited to a rate of 40 Kbytes per second (with an allowed burst of 50 Kbytes). Note that the rate is specified in bytes, not bits, per second; a rate of 40000 bytes per second, nominally corresponds to 320 kbps.

Rule #3: traffic from the subnet 10.31.4.xxx destined to ports 4000-9000 is always blocked, regardless of protocol.

Rule #4: traffic from the subnet 10.4.xxx.yyy destined to ports 4000-9000 is always blocked, regardless of protocol.

All other traffic is allowed

To edit a rule:

1. In the 'Edit Rule' column, select the rule that you want to edit.

Modify the fields as desired.

Click the **Apply** button to save the changes.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

To activate a de-activated rule:

1. In the 'Edit Rule' column, select the de-activated rule that you want to activate.

Click the **Activate** button; the rule is activated.

To de-activate an activated rule:

1. In the 'Edit Rule' column, select the activated rule that you want to de-activate.

Click the **DeActivate** button; the rule is de-activated.

To delete a rule:

1. Select the radio button of the entry you want to activate.

Click the **Delete Rule** button; the rule is deleted.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

Table 12-1: Internal Firewall Parameters

Parameter	Description
Rule Status	A read-only field indicating whether the rule is active or not. **Note:** After device reset, all rules are active.
Source IP **[AccessList_Source_IP]**	IP address (or DNS name) or a specific host name of the source network (i.e., from where the incoming packet is received).
Source Port **[AccessList_Source_Port]**	Defines the source UDP/TCP ports (on the remote host) from where packets are sent to the device. The valid range is 0 to 65535. **Note:** When set to 0, this field is ignored and any source port matches the rule.
Prefix Length **[AccessList_PrefixLen]**	IP network mask. 32 for a single host, or the appropriate value for the source IP addresses. A value of 8 corresponds to IPv4 subnet class A (network mask of 255.0.0.0). A value of 16 corresponds to IPv4 subnet class B (network mask of 255.255.0.0). A value of 24 corresponds to IPv4 subnet class C (network mask of 255.255.255.0). The IP address of the sender of the incoming packet is trimmed in accordance with the prefix length (in bits) and then compared to the parameter 'Source IP'.
Source Port **[AccessList_Source_Port]**	Defines the source UDP or TCP ports (on the remote host) from where packets are sent to the device. The valid range is 0 to 65535. **Note:** When set to 0, this field is ignored and any port matches the rule.
Local Port Range **[AccessList_Start_Port]** **[AccessList_End_Port]**	The destination UDP/TCP ports (on this device) to which packets are sent. The valid range is 0 to 65535. **Note:** When the protocol type isn't TCP or UDP, the entire range must be provided.
Protocol **[AccessList_Protocol]**	The protocol type (e.g., UDP, TCP, ICMP, ESP or 'Any'), or the IANA protocol number (in the range of 0 (Any) to 255). **Note:** This field also accepts the abbreviated strings 'SIP' and 'HTTP'. Specifying these strings implies selection of the TCP or UDP protocols, and the appropriate port numbers as defined on the device.
Use Specific Interface **[AccessList_Use_Specific_Interface]**	Determines whether you want to apply the rule to a specific network interface defined in the Multiple Interface table (i.e., packets received from that defined in the Source IP field and received on this network interface): **[0]** Disable (default) **[1]** Enable **Notes:** If enabled, then in the 'Interface Name' field (described below), select the interface to which the rule is applied. If disabled, then the rule applies to all interfaces.
Interface Name	The network interface to which you want to apply the rule. This is

Parameter	Description
[AccessList_Interface_ID]	applicable if you enabled the 'Use Specific Interface' field. The list displays interface names as defined in the Multiple Interface table (see 'Configuring IP Interface Settings' on page 68).
Packet Size **[AccessList_Packet_Size]**	Maximum allowed packet size. The valid range is 0 to 65535. **Note:** When filtering fragmented IP packets, this field relates to the overall (re-assembled) packet size, and not to the size of each fragment.
Byte Rate **[AccessList_Byte_Rate]**	Expected traffic rate (bytes per second). This field defines the allowed bandwidth for the specified protocol. In addition to this field, the 'Burst Bytes' field provides additional allowance such that momentary bursts of data may utilize more than the defined byte rate, without being interrupted. For example, if 'Byte Rate' is set to 40000 and 'Burst Bytes' to 50000, then this implies the following: the allowed bandwidth is 40000 bytes/sec with extra allowance of 50000 bytes; if, for example, the actual traffic rate is 45000 bytes/sec, then this allowance would be consumed within 10 seconds, after which all traffic exceeding the allocated 40000 bytes/sec is dropped. If the actual traffic rate then slowed to 30000 bytes/sec, then the allowance would be replenished within 5 seconds.
Burst Bytes **[AccessList_Byte_Burst]**	Tolerance of traffic rate limit (number of bytes).
Action Upon Match **[AccessList_Allow_Type]**	Action upon match (i.e., 'Allow' or 'Block').
Match Count **[AccessList_MatchCount]**	A read-only field displaying the number of packets accepted/rejected by the specific rule.

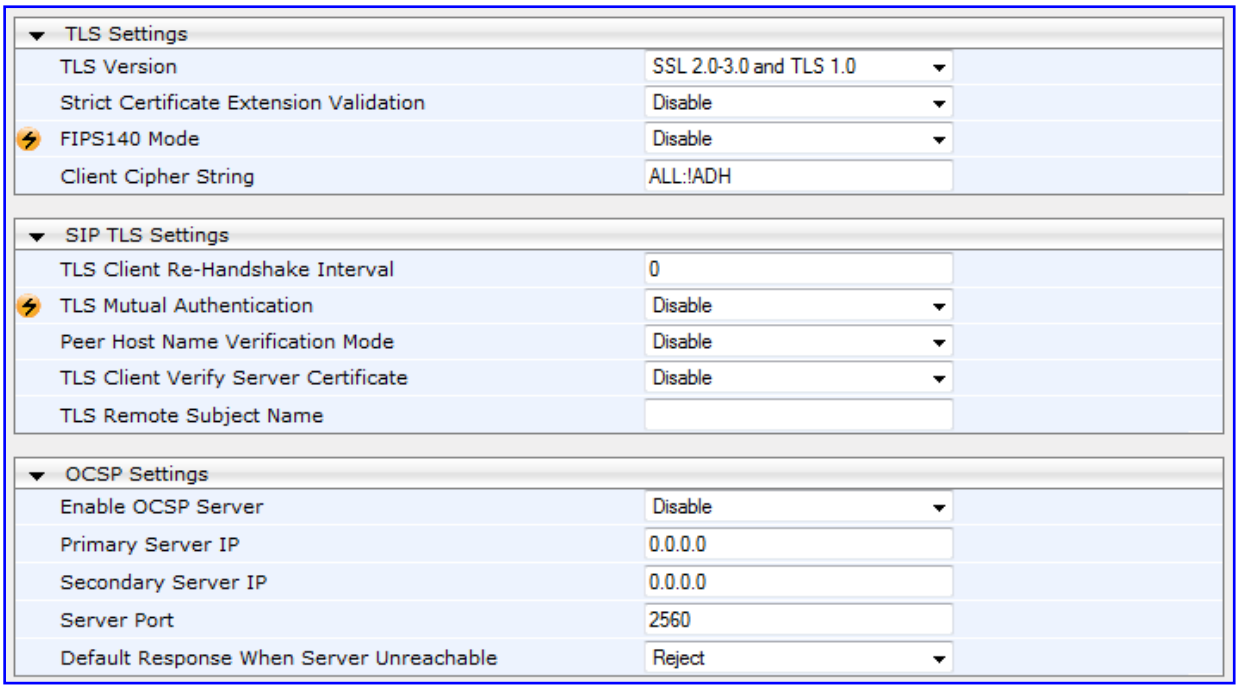
12.2 Configuring General Security Settings

The General Security Settings page is used to configure various security features. For a description of the parameters appearing on this page, refer 'Configuration Parameters Reference' on page **Erreur ! Signet non défini.**.

To configure the general security parameters:

1. Open the General Security Settings page (**Configuration** tab >**VoIP** menu >**Security** submenu >**General Security Settings**).

Figure 12-2: General Security Settings Page



Configure the parameters as required.

Click Subm**it to** apply your changes.

To save the changes to flash memory, refer to 'Saving Configuration' on page 245.

13 Media

This section describes the media-related configuration.

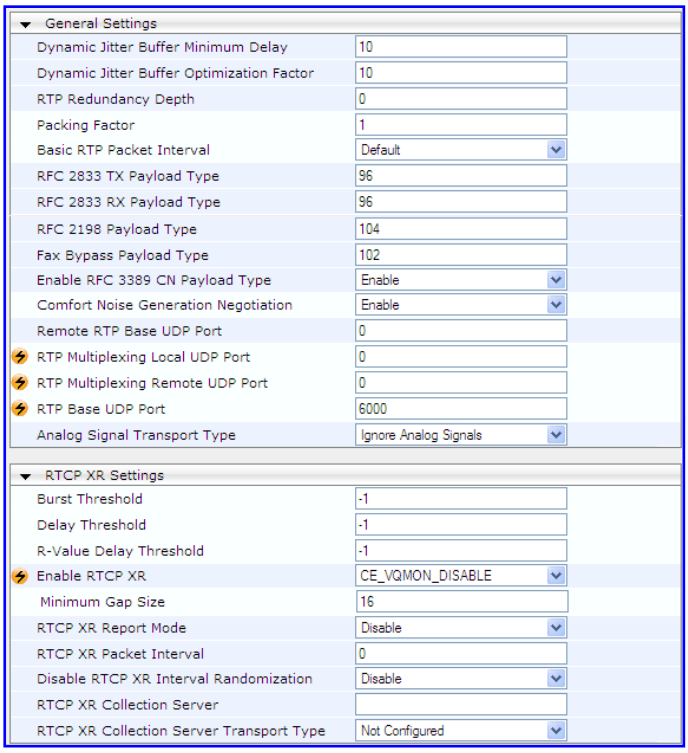
13.1 Configuring RTP/RTCP Settings

The RTP/RTCP Settings page configures the Real-Time Transport Protocol (RTP) and Real-Time Transport (RTP) Control Protocol (RTCP) parameters. For a detailed description of the parameters appearing on this page, refer to 'Configuration Parameters Reference' on page **Erreur ! Signet non défini.**.

To configure the RTP/RTCP parameters:

1. Open the RTP/RTCP Settings page (**Configuration** tab >**VoIP** menu >**Media** submenu >**RTP/RTCP Settings**).

Figure 13-1: RTP/RTCP Settings Page



Note: Many of the parameters are on this page are currently not supported. For more information, contact your sales representative.

Configure the parameters as required.

Click Subm**it to** apply your changes.

To save the changes to flash memory, refer to 'Saving Configuration' on page 245.

13.1.1 Configuring RTP Base UDP Port

You can configure the range of UDP ports for RTP, RTCP, and T.38. The UDP port range can be configured using media realms in the Media Realm table, allowing you to assign different port ranges (media realms) to different interfaces. However, if you do not use media realms, you can configure the lower boundary of the UDP port used for RTP, RTCP (RTP port + 1) and T.38 (RTP port + 2), using the 'RTP Base UDP Port' (BaseUDPport) parameter. For example, if the Base UDP Port is set to 6000, then one channel may use the ports RTP 6000, RTCP 6001, and T.38 6002, while another channel may use RTP 6010, RTCP 6011, and T.38 6012. The range of possible UDP ports is 6,000 to 64,000 (default base UDP port is 6000).

The port range is calculated using the 'RTP Base UDP Port' (BaseUDPport) parameter as follows: **BaseUDPPort to (BaseUDPPort + <channels -1> * 10)**

The maximum (when all channels are required) UDP port range is calculated as follows:**BaseUDPport to BaseUDPport + 4000*10**

Notes:

The device allocates the UDP ports randomly to the channels.

If you are using Media Realms (see 'Configuring Media Realms' on page 105), the port range configured for the Media Realm must be within this range defined by the BaseUDPPort parameter.

13.1.1.1 Configuring RTP Base UDP Port using Web Interface

The procedure below describes how to configure the RTP base UDP port using the Web interface.

To configure the RTP base UDP port:

1. Open the RTP/RTCP Settings page (**Configuration** tab >**VoIP** menu >**Media** submenu >**RTP/RTCP Settings**).

Set the 'RTP Base UDP Port' parameter to the required value.

Click **Submit**.

Reset the device for the settings to take effect.

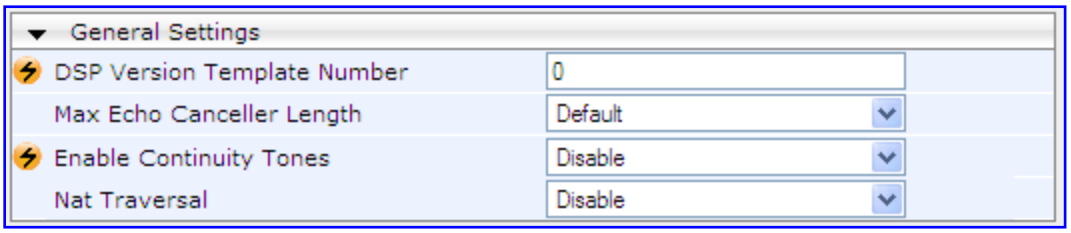
13.2 Configuring General Media Settings

The General Media Settings page allows you to configure various media parameters. For a detailed description of the parameters appearing on this page, see 'Configuration Parameters Reference' on page **Erreur ! Signet non défini.**.

To configure general media parameters:

1. Open the General Media Settings page (**Configuration** tab >**VoIP** menu >**Media** submenu >**General Media Settings**).

Figure 13-2: General Media Settings Page



Configure the parameters as required.

Click Subm**it to** apply your changes.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

13.3 Configuring Media Realms

The Media Realm Table page allows you to define a pool of up to 64 SIP media interfaces, termed *Media Realms*. Media Realms allow you to divide a Media-type interface (defined in the Multiple Interface table - see 'Configuring IP Interface Settings' on page 68) into several realms, where each realm is specified by a UDP port range. In addition, you can define the maximum number of sessions per Media Realm. Once created, Media Realms can be assigned to IP Groups (in the IP Group table - see 'Configuring IP Groups' on page 114) or SRDs (in the SRD table - see 'Configuring SRD Table' on page 110).

For each Media Realm you can configure Quality of Experience parameters and their thresholds for reporting to the Alcatel-Lucent SEM server used for monitoring the quality of calls. For configuring this, see 'Configuring Quality of Experience Parameters per Media Realm' on page 108.

Notes:

If different Media Realms are assigned to an IP Group and to an SRD, the IP Group's Media Realm takes precedence.

For this setting to take effect, a device reset is required.

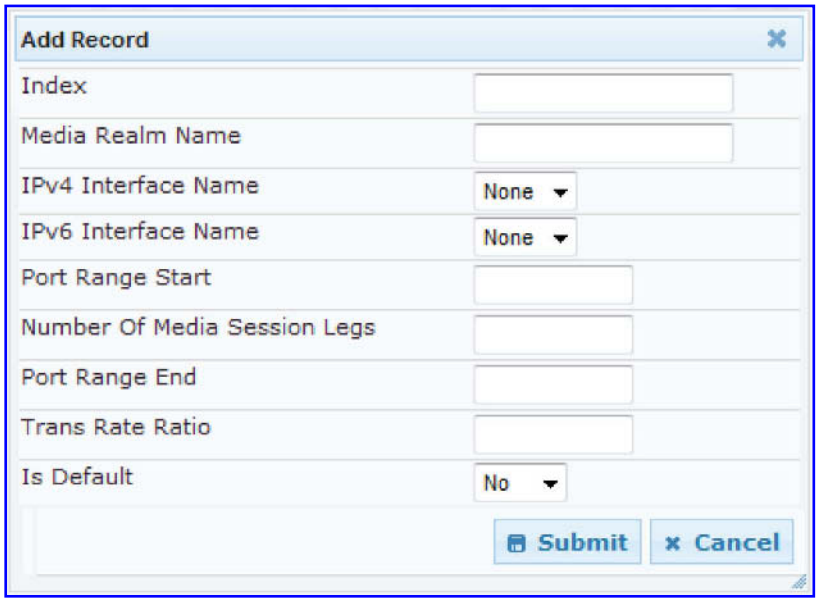
You can also configure the Media Realm table using the *ini* file table parameter CpMediaRealm.

To define a Media Realm:

1. Open the Media Realm Table page (**Configuration** tab >**VoIP** menu >**Media** submenu >**Media Realm Configuration**).

Click the **Add** button; the following appears:

Figure 13-3: Add Record Dialog Box



Configure the parameters as required. See the table below for a description of each parameter

Click **Submit** to apply your settings.

Reset the device to save the changes to flash memory (see 'Saving Configuration' on page 245).

Table 13-1: Media Realm Table Parameter Descriptions

Parameter	Description
Index **[CpMediaRealm_Index]**	Defines the required table index number.
Media Realm Name **[CpMediaRealm_MediaRealmName]**	Defines an arbitrary, identifiable name for the Media Realm. The valid value is a string of up to 40 characters. **Notes:** This parameter is mandatory. The name assigned to the Media Realm must be unique. This Media Realm name is used in the SRD and IP Groups table.
IPv4 Interface Name **[CpMediaRealm_IPv4IF]**	Assigns an IPv4 interface to the Media Realm. **Note:** The name of this interface must be identical (i.e., case-sensitive etc.) as configured in the Multiple Interface table (InterfaceTable parameter).
IPv6 Interface Name **[CpMediaRealm_IPv6IF]**	Assigns anIPv6 interface to the Media Realm. **Note:** The name of this interface must be identical as configured in the Multiple Interface table (InterfaceTable parameter).

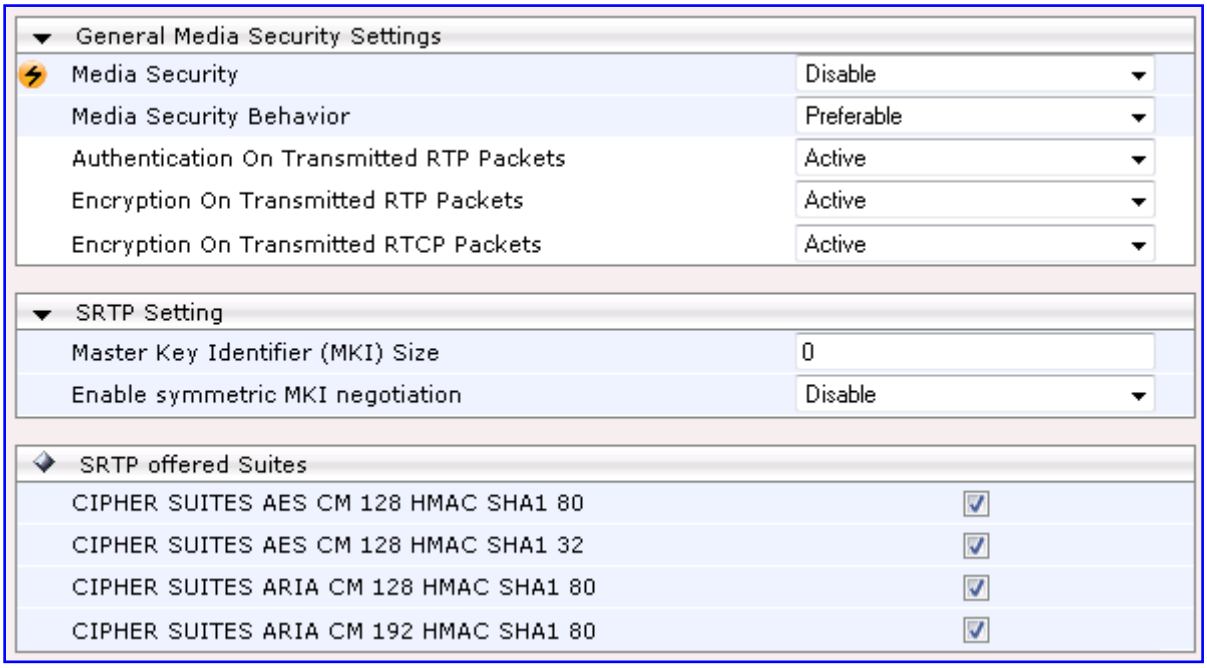
Parameter	Description
Port Range Start **[CpMediaRealm_PortRangeStart]**	Defines the starting port for the range of Media interface UDP ports. **Notes:** You must either configure all media realms with port ranges or without (not some with and some without). The available UDP port range is calculated using the BaseUDPport parameter: BaseUDPport to BaseUDPport + 4000*10 Port ranges over 60,000 must not be used. Ranges of Media Realm ports must not overlap.
Number of Media Session Legs **[CpMediaRealm_MediaSessionLeg]**	Defines the number of media sessions associated with the range of ports. This is the number of media sessions available in the port range. For example, 100 ports correspond to 10 media sessions, since ports are allocated in chunks of 10.
Port Range End **[CpMediaRealm_PortRangeEnd]**	Read-only field displaying the ending port for the range of Media interface UDP ports. This field is calculated by adding the 'Media Session Leg' field (multiplied by the port chunk size) to the 'Port Range Start' field. A value appears once a row has been successfully added to the table.
Trans Rate Ratio **[CpMediaRealm_TransRateRatio]**	**Note:** This field will be supported in the next applicable release.
Is Default **[CpMediaRealm_IsDefault]**	Defines the Media Realm as the default Media Realm. This default Media Realm is used when no Media Realm is configured for an IP Group or SRD for a specific call. **[0]** No (default) **[1]** Yes **Notes:** This parameter can be set to Yes (1) for only **one** defined Media Realm. If this parameter is not configured, then the first Media Realm in the table is used as default. If the table is not configured, then the default Media Realm includes all the configured media interfaces.

13.4 Configuring Media Security

The Media Security page allows you to configure media security. For a detailed description of the parameters appearing on this page, see 'Configuration Parameters Reference' on page **Erreur ! Signet non défini.**.

To configure media security:

1. Open the Media Security page (**Configuration** tab >**VoIP** menu >**Media** submenu >**Media Security**).



Configure the parameters as required.

Click Subm**it to** apply your changes.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

14 Enabling Applications

The device supports the following main applications:

Stand-Alone Survivability (SAS) application

Session Border Control (SBC) application

The procedure below describes how to enable these applications. Once an application is enabled, the Web GUI provides menus and parameter fields relevant to the application.

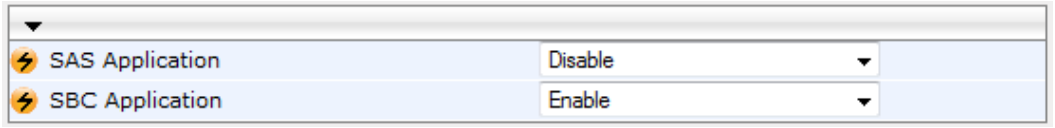
Notes:

- This page displays the application only if the device is installed with the relevant Software Upgrade Key supporting the application (see 'Loading Software Upgrade Key' on page 250).
- For configuring the SAS application, see 'Stand-Alone Survivability (SAS) Application' on page 201.
- For configuring the SBC application, see Session Border Controller on page 139.
- For enabling an application, a device reset is required.

To enable an application:

1. Open the Applications Enabling page (**Configuration** tab >**VoIP** menu >**Applications Enabling** submenu >**Applications Enabling**).

Figure 14-1: Applications Enabling Page



From the relevant application drop-down list, select **Enable**.

Save (burn) the changes to the device's flash memory with a device reset (see 'Saving Configuration' on page 245).

15 Control Network

This section describes configuration of the network at the SIP control level.

15.1 Configuring SRD Table

The SRD Settings page allows you to configure up to 32 signaling routing domains (SRD). An SRD is configured with a unique name and assigned a Media Realm (defined in the Media Realm table - see 'Configuring Media Realms' on page 105). In addition, other SBC attributes such as media anchoring and user registration can also be configured. Once configured, you can use the SRDs as follows:

Associate it with a SIP Interface (see 'Configuring SIP Interface Table' on page 112)

Associate it with an IP Group (see Configuring IP Groups on page 114)

Associate it with a Proxy Set (see Configuring Proxy Sets Table on page 120)

Apply an Admission Control rule to it (see Configuring Admission Control Table on page 173)

Define it as a Classification rule for the incoming SIP request (see Configuring Classification Table on page 178)

Use it as a destination IP-to-IP routing rule (see 'Configuring IP-to-IP Routing Table' on page 183)

Therefore, an SRD is a set of definitions together creating multiple, virtual multi-service IP gateways:

Multiple and different SIP signaling interfaces (SRD associated with a SIP Interface) and RTP media (associated with a Media Realm) for multiple Layer-3 networks. Due to the B2BUA nature of the SBC application, different interfaces can be assigned to each leg of the call.

Can operate with multiple gateway customers that may reside either in the same or in different Layer-3 networks as the device. This allows separation of signaling traffic between different customers. In such a scenario, the device is configured with multiple SRD's.

Typically, one SRD is defined for each group of SIP UAs (e.g. proxies, IP phones, application servers, gateways, and softswitches) that communicate with each other. This provides these entities with VoIP services that reside on the same Layer-3 network (must be able to communicate without traversing NAT devices and must not have overlapping IP addresses). Routing from one SRD to another is possible, whereby each routing destination (IP Group or destination address) indicates the SRD to which it belongs.

The SRD Settings page also displays the IP Groups, Proxy Sets, and SIP Interfaces associated with a selected SRD index.

Notes:

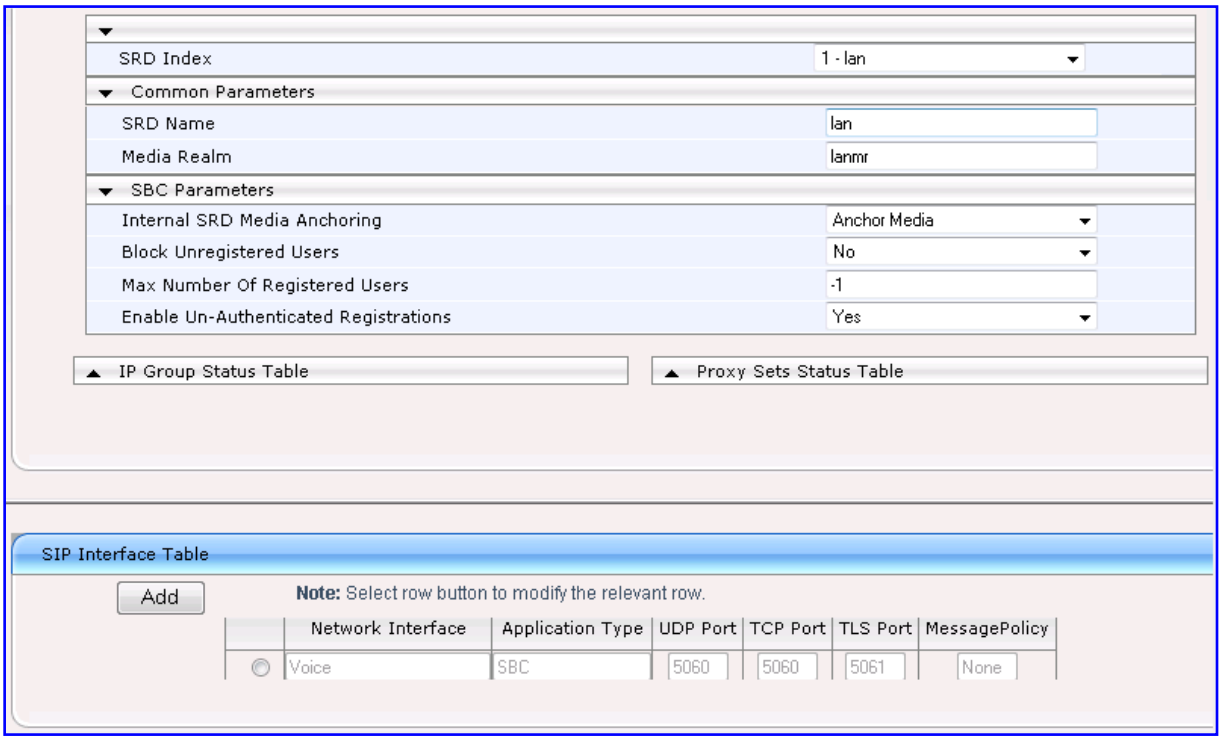
For a detailed description of SRD's, see 'Multiple SIP Signaling/Media Interfaces Environment' on page 127.

The SRD table can also be configured using the *ini* file table parameter SRD.

To configure SRDs:

1. Open the SRD Settings page (**Configuration** tab >**VoIP** menu >**Control Network** submenu >**SRD Table**).

Figure 15-1: SRD Settings Page



From the 'SRD Index' drop-down list, select an index for the SRD, and then configure it according to the table below.

Click Subm**it to** apply your changes.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

Note: The SRD Settings page also allows you to define a SIP Interface in the SIP Interface table, instead of navigating to the SIP Interface Table page as described in 'Configuring SIP Interface Table' on page 112.

Table 15-1: SRD Table Parameters

Parameter	Description
SRD Name **[SRD_Name]**	Mandatory descriptive name of the SRD. The valid value can be a string of up to 21 characters.
Media Realm **[SRD_MediaRealm]**	Defines the Media Realm associated with the SRD. The entered string value must be identical (including case-sensitive) to the Media Realm name as defined in the Media Realm table. The valid value is a string of up to 40 characters. **Notes:** If the Media Realm is later deleted from the Media Realm table, then this value becomes invalid in the SRD table. For configuring Media Realms, see 'Configuring Media Realms' on page 105.

Parameter	Description
Internal SRD Media Anchoring **[SRD_IntraSRDMediaAnchoring]**	Determines whether the device performs media anchoring or not on media for the SRD. **[0]** Anchor Media (default) = RTP traverses the device and each leg uses a different coder or coder parameters. **[1]** Don't Anchor Media = The RTP packet flow does not traverse the device; instead, the two SIP UA's establish a direct RTP/SRTP (media) flow between one another. **Notes:** When No Media Anchoring is enabled: The device does not perform manipulation on SDP data (offer/answer transactions) such as ports, IP address, and coders. Opening voice channels and allocation of IP media ports are not required. When two UA's pertain to the same SRD and this parameter is set to **[1],** and one of the UA's is defined as a foreign user (example, “follow me service”) located on the WAN while the other UA is located on the LAN, then calls between these two UA's can’t be established until this parameter is set to 0, as the device doesn’t interfere in the SIP signaling. In other words, parameters such as IP addresses are not manipulated for calls between LAN and WAN (although required). When the global parameter SBCDirectMedia is disabled, you cannot enable No Media Anchoring for two UA's pertaining to separate SRDs; No Media Anchoring can only be enable for two UA's pertaining to the same SRD. For more information on media handling, see SBC Media Handling on page 154.
Block Unregistered Users **[SRD_BlockUnRegUsers]**	Determines whether the device blocks (rejects) incoming calls (INVITE requests) from unregistered users (pertaining to USER-type IP Groups) for the SRD. **[0]** No = Calls from unregistered users are not blocked (default). **[1]** Yes = Blocks calls from unregistered users. **Note:** When the call is blocked, the device sends a SIP 500 "Server Internal Error" response to the remote end.
Max Number of Registered Users **[SRD_MaxNumOfRegUsers]**	Maximum number of users belonging to this SRD that can register with the device. By default, no limitation exists for registered users
Enable Un-Authenticated Registrations **[SRD_EnableUnAuthenticatedRegistrations]**	Determines whether the device blocks REGISTER requests from new users (i.e., users not registered in the device's registration database) when the destination IP Group is of type USER. **[0]** No = The device sends REGISTER requests to the SIP proxy server and only if authenticated by the server does the device add the user registration to its database. **[1]** Yes = The device adds REGISTER requests to its database even if the requests are not authenticated by a SIP proxy (default).

15.2 Configuring SIP Interface Table

The SIP Interface Table page allows you to configure up to 32 SIP signaling interfaces, referred to as *SIP Interfaces*. A SIP Interface consists of a combination of ports (UDP, TCP, and TLS),

associated with a specific IP address (IPv4 / IPv6) and for a specific application (i.e., SAS and SBC). Once defined, the SIP Interface can then be associated with an SRD (in the SRD Settings page - see 'Configuring SRD Table' on page 110).

SIP Interfaces can be used for the following:

Implementing SIP signaling interfaces for each call leg (i.e., each SIP UA communicates with a specific SRD).

Implementing different SIP signaling ports (listening UDP, TCP, and TLS, and the UDP source ports) for a single interface or for multiple interfaces.

Differentiating between applications (i.e., SAS and SBC) by creating SIP Interfaces per application.

Separating signaling traffic between networks (e.g., different customers) to use different routing tables, manipulations, SIP definitions, and so on.

Notes:

The SIP Interface table also appears in the SRD Settings page, allowing you to add SIP Interfaces there as well.

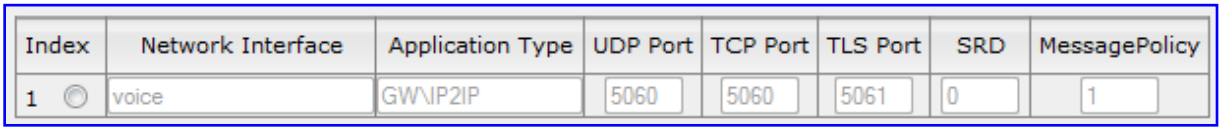
For more information on SIP interfaces, see 'Multiple SIP Signaling/Media Interfaces Environment' on page 127.

The SIP Interface table can also be configured using the *ini* file table parameter SIPInterface.

To configure the SIP Interface table:

1. Open the SIP Interface Table page (**Configuration** tab >**VoIP** menu >**Control Network** submenu >**SIP Interface Table**).

Figure 15-2: SIP Interface Table Page



Add an entry and then configure it according to the table below.

Click the **Apply** button to save your changes.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

Table 15-2: SIP Interface Table Parameters

Parameter	Description
Network Interface **[SIPInterface_NetworkInterface]**	Defines the Control-type IP network interface that you want to associate with the SIP Interface. This value string must be identical (including case-sensitive) to that configured in the 'Interface Name' in the Multiple Interface table (see 'Configuring IP Interface Settings' on page 68). The default is "Not Configured". **Note:** SIP Interfaces that are assigned to a specific SRD must be defined with the same network interface. For example, if you define three SIP Interfaces for SRD ID #8, all these SIP Interfaces must be defined with the same network interface (e.g., "SIP1").
Application Type **[SIPInterface_ApplicationType]**	Defines the application type associated with the SIP Interface. **[1]** SAS = Stand-Alone Survivability (SAS) application **[2]** SBC = SBC application
UDP Port	Defines the listening and source UDP port.

Parameter	Description
[SIPInterface_UDPPort]	The valid range is 1 to 65534. The default is 5060. **Notes:** This port must be outside of the RTP port range. Each SIP Interface must have a unique signaling port (i.e., no two SIP Interfaces can share the same port - no port overlapping).
TCP Port **[SIPInterface_TCPPort]**	Defines the listening TCP port. The valid range is 1 to 65534. The default is 5060. **Notes:** This port must be outside of the RTP port range. Each SIP Interface must have a unique signaling port (i.e., no two SIP Interfaces can share the same port - no port overlapping).
TLS Port **[SIPInterface_TLSPort]**	Defines the listening TLS port. The valid range is 1 to 65534. The default is 5061. **Notes:** This port must be outside of the RTP port range. Each SIP Interface must have a unique signaling port (i.e., no two SIP Interfaces can share the same port - no port overlapping).
SRD **[SIPInterface_SRD]**	Defines the SRD ID associated with the SIP Interface. The default SRD is 0. **Notes:** Each SRD can be associated with up to three SIP Interfaces, where each SIP Interface pertains to a different Application Type (GW/IP2IP, SAS, and SBC). SIP Interfaces that are assigned to a specific SRD must be defined with the same network interface. For example, if you define three SIP Interfaces for SRD ID #8, all these SIP Interfaces must be defined with the same network interface (e.g., "SIP1"). To configure SRDs, see 'Configuring SRD Table' on page 110.
Message Policy **[SIPInterface_MessagePolicy]**	Assigns a SIP message policy to the SIP interface. **Note:** To configure SIP message policies, see Configuring SIP Message Policy Rules.

15.3 Configuring IP Groups

The IP Group Table page allows you to create up to 32 logical IP entities called *IP Groups*. An IP Group is an entity with a set of definitions such as a Proxy Set ID (see 'Configuring Proxy Sets Table' on page 120), which represents the IP address of the IP Group.

IP Groups provide the following uses:

SIP dialog registration and authentication (digest user/password) of a specific IP Group (*Served* IP Group, e.g., corporate IP-PBX) with another IP Group (*Serving* IP Group, e.g., ITSP). This is configured in the Account table (see 'Configuring Account Table' on page 132).

For the SBC application, IP Groups are used to classify incoming SIP dialog-initiating requests (e.g., INVITE messages) to a source IP Group, based on Proxy Set ID (defined in Configuring Classification Table on page 178). This occurs if the database search for a registered user is unsuccessful. The classification process locates a Proxy Set ID (associated with the SIP dialog request's IP address) in the Proxy Set table, and then locates a match with an IP Group that is associated with this Proxy Set in the IP Group table. This classification is enabled using the parameter Classify By Proxy Set.

Notes:

When operating with multiple IP Groups, the default Proxy server must not be used (i.e., the parameter IsProxyUsed must be set to 0).

If different SRDs are configured in the IP Group and Proxy Set tables, the SRD defined for the Proxy Set takes precedence.

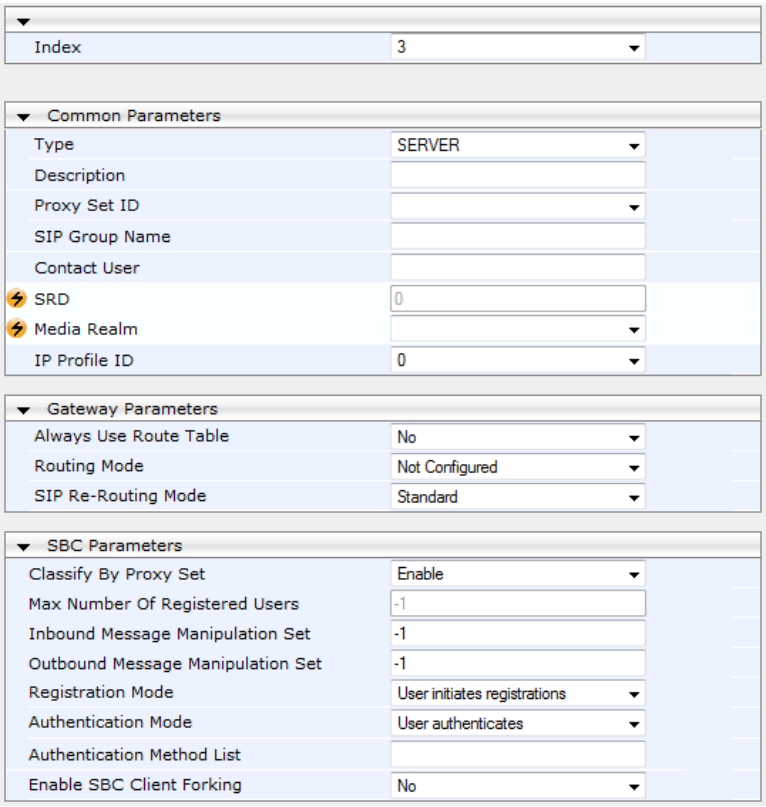
You cannot modify IP Group index 0. This IP Group is set to default values and is used by the device when IP Groups are not implemented.

You can also configure the IP Groups table using the *ini* file table parameter IPGroup (see 'Configuration Parameters Reference' on page **Erreur ! Signet non défini.**).

To configure IP Groups:

1. Open the IP Group Table page (**Configuration** tab >**VoIP** menu >**Control Network** submenu >**IP Group Table**).

Figure 15-3: IP Group Table



Configure the IP group parameters according to the table below.

Click Subm**it to** apply your changes.

To save the changes to flash memory, see 'Saving Configuration' **on page 245.**

Table 15-3: IP Group Parameters

Parameter	Description
Common Parameters	
Type **[IPGroup**_Type]	The IP Group can be defined as one of the following types: **[0]** SERVER = used when the destination address (configured by the Proxy Set) of the IP Group (e.g., ITSP, Proxy, IP-PBX, or Application server) is known. **[1]** USER = represents a group of users (such as IP phones and softphones) where their location is dynamically obtained by the device when REGISTER requests and responses traverse (or are terminated) by the device. These users are considered remote (far-end) users. Typically, this IP Group is configured with a Serving IP Group that represents an IP-PBX, Application or Proxy server that serves this USER-type IP Group. Each SIP request sent by a user of this IP Group is proxied to the Serving IP Group. For registrations, the device updates its internal database with the AOR and contacts of the users. Digest authentication using SIP 401/407 responses (if needed) is performed by the Serving IP Group. The device forwards these responses directly to the SIP users. To route a call to a registered user, a rule must be configured in the IP2IP Routing Table table (see 'Configuring SBC IP-to-IP Routing' on page 183). The device searches the dynamic database (by using the request URI) for an entry that matches a registered AOR or Contact. Once an entry is found, the IP destination is obtained from this entry, and a SIP request is sent to the destination.The device also supports NAT traversal for the SIP clients that are behind NAT. In this case, the device must be defined with a global IP address. **[2]** GATEWAY = This is applicable only to the SBC application in scenarios where the device receives requests to and from a gateway representing multiple users. This IP Group type is necessary as the other IP Group types are not suitable: The IP Group cannot be defined as a SERVER since its destination address is unknown during configuration. The IP Group cannot be defined as a USER since the SIP Contact header of the incoming REGISTER does not represent a specific user. The Request-URI user part can change and therefore, the device is unable to identify an already registered user and therefore, adds an additional record to the database. The IP address of the “GATEWAY” IP Group is obtained dynamically from the host part of the Contact header in the REGISTER request received from the IP Group. Therefore, routing to this IP Group is possible only once a REGISTER request is received. If a REGISTER refresh request arrives, the device updates the new location (i.e., IP address) of the IP Group. If the REGISTER fails, no update is performed. If an UN-REGISTER request arrives, the IP address associated with the IP Group is deleted and therefore, no routing to the IP Group is done. **Note:** This field is available only if the SBC application is enabled.
Description	Brief string description of the IP Group. The value range is a string of up to 29 characters. The default is

Parameter	Description
[IPGroup_Description]	an empty field.
Proxy Set ID **[IPGroup_ProxySetId]**	The Proxy Set ID (defined in 'Configuring Proxy Sets Table' on page 120) associated with the IP Group. All INVITE messages destined to this IP Group are sent to the IP address associated with the Proxy Set. **Notes:** Proxy Set ID 0 must **not** be selected; this is the device's default Proxy. The Proxy Set is applicable only to SERVER-type IP Groups.
SIP Group Name **[IPGroup_SIPGroupName]**	The SIP Request-URI host name used in INVITE and REGISTER messages sent to the IP Group, or the host name in the From header of INVITE messages received from the IP Group. If not specified, the value of the global parameter, ProxyName (see 'Configuring Proxy and Registration Parameters' on page 135) is used instead. The value range is a string of up to 100 characters. The default is an empty field. **Note:** If the IP Group is of type USER, this parameter is used internally as a host name in the Request-URI. For example, if an incoming call is routed to a USER-type IP Group, the device first creates the Request-URI (<destination_number>@<SIP Group Name>), and then it searches the user's internal database for a match.
Contact User **[IPGroup_ContactUser]**	Defines the user part for the From, To, and Contact headers of SIP REGISTER messages, and the user part for the Contact header of INVITE messages that are received from the IP Group and forwarded by the device to another IP Group. **Notes:** This parameter is applicable only to SERVER-type IP Groups. This parameter is overridden by the 'Contact User' parameter in the 'Account' table (see 'Configuring Account Table' on page 132).
Domain Name in Contact **[IPGroup_ContactName]**	Defines the host name (string) that the device uses in the SIP message's Via and Contact headers. This is typically used to define an FQDN as the host name. The device uses this string for Via and Contact headers in outgoing INVITE messages to a specific IP Group, and the Contact header in SIP 18x and 200 OK responses for incoming INVITE messages from a specific IP Group. The Inbound IP Routing table can be used to identify the source IP Group from where the INVITE message was received. If this parameter is not configured (default), these headers are populated with the device's dotted-decimal IP address of the network interface on which the message is sent. **Note:** To ensure proper device handling, this parameter should be a valid FQDN.
SRD **[IPGroup**_SRD]	The SRD (defined in Configuring SRD Table on page 110) associated with the IP Group. The default is 0. **Note:** For this parameter to take effect, a device reset is required.
Media Realm	Assigns a Media Realm to the IP Group. The entered string value must be identical (including case-sensitive) to the Media Realm

Parameter	Description
[IPGroup_MediaRealm]	name as defined in the Media Realm table. **Notes:** For this parameter to take effect, a device reset is required. If the Media Realm is later deleted from the Media Realm table, then this value becomes invalid. For configuring Media Realms, see Configuring Media Realms on page 105.
IP Profile ID **[IPGroup_ProfileId]**	The IP Profile (defined in to 'Configuring IP Profile Settings' on page 137) that you want assigned to this IP Group. The default is 0.
SBC Parameters	
Classify By Proxy Set **[IPGroup**_ClassifyByProxySet]	Determines whether the incoming INVITE is classified to an IP Group according to the Proxy Set. **[0]** Disable **[1]** Enable (default) This classification occurs only if classification according to the device's database fails for locating whether the INVITE arrived from a registered user. The classification proceeds with checking whether the INVITE's IP address (if host names, then according to the dynamically resolved IP address list) is defined in the IP Group's Proxy Set ID (in the Proxy Set table). If the IP address is listed, then the INVITE is assigned to this IP Group. **Notes:** This parameter is applicable only to SERVER-type IP Groups. This classification is not relevant in cases where multiple IP Groups use the same Proxy Set.
Max Number Of Registered Users **[IPGroup**_MaxNumOfRegUsers]	Maximum number of users belonging to this IP Group that can register with the device. By default, no limitation exists for registered users. **Note:** This field is applicable only to USER-type IP Groups.
Inbound Message Manipulation Set **[IPGroup**_InboundManSet]	Message Manipulation Set (rule) that you want to assign to this IP Group for SIP message manipulation rule on the inbound message. The Message Manipulation rules are configured using the MessageManipulations parameter (see Configuring Message Manipulations on page 192).
Outbound Message Manipulation Set **[IPGroup**_OutboundManSet]	Message Manipulation Set (rule) that you want to assign to this IP Group for SIP message manipulation on the outbound message. The Message Manipulation rules are configured using the MessageManipulations parameter (see Configuring Message Manipulations on page 192).
Registration Mode **[IPGroup**_RegistrationMode]	Defines the registration mode for an IP Group: **[0]** User initiates registrations (default) **[1]** SBC initiate registrations (works only with User Info file) = Used when the device serves as a client (e.g., with an IP PBX). **[2]** No registrations needed = The device adds users to its database in active state.

Parameter	Description
Authentication Mode **[PGroup**_AuthenticationMode]	Defines the authentication mode. **[0]** User Authenticates (default) = The device does not handle the authentication, but simply passes the authentication messages between the SIP user agents. **[1]** SBC Authenticates (as client) = The device authenticates as a client. It receives the 401/407 response from the proxy requesting for authentication. The device sends the proxy the authorization credentials (i.e., user name and password) according to one of the following: 1) account defined in the Account table (only if authenticating SERVER-type IP Group), 2) User Information file, or 3) sends request to users requesting credentials (only if authenticating USER-type IP Group). **[2]** SBC Authenticates (as server) = The device authenticates as a server (using the User Information file).
Authentication Method List **[IPGroup**_MethodList]	Defines SIP methods that the device must challenge. Multiple entries are separated by the forward slash "/". If none are defined (default), no methods are challenged.
Enable SBC Client Forking **[IPGroup**_EnableSBCClientForking]	Enables call forking for USER-type IP Groups. **[0]** No (default) **[1]** Yes = The device forks INVITE messages (to up to five separate SIP outgoing legs).

15.4 Configuring Proxy Sets Table

The Proxy Sets Table page allows you to define *Proxy Sets*. A Proxy Set is a group of Proxy servers defined by IP address or fully qualified domain name (FQDN). You can define up to 32 Proxy Sets, each with a unique ID number and up to five Proxy server addresses. For each Proxy server address you can define the transport type (i.e., UDP, TCP, or TLS). In addition, Proxy load balancing and redundancy mechanisms can be applied per Proxy Set (if a Proxy Set contains more than one Proxy address).

Proxy Sets can later be assigned to IP Groups of type SERVER (see 'Configuring IP Groups' on page 114). When the device sends an INVITE message to an IP Group, it is sent to the IP address or domain name defined for the Proxy Set that is associated with the IP Group. In other words, the Proxy Set represents the **destination** of the call. Typically, for IP-to-IP call routing, at least two Proxy Sets are defined for call destination – one for each leg (IP Group) of the call (i.e., both directions). For example, one Proxy Set for the Internet Telephony Service provider (ITSP) interfacing with one 'leg' of the device and another Proxy Set for the second SIP entity (e.g., ITSP) interfacing with the other 'leg' of the device.

Notes:

You can also configure the Proxy Sets table using two complementary *ini* file table parameters (see 'Configuration Parameters Reference' on page **Erreur ! Signet non défini.**):
- ProxyIP: used for creating a Proxy Set ID defined with IP addresses.
- ProxySet: used for defining various attributes for the Proxy Set ID.

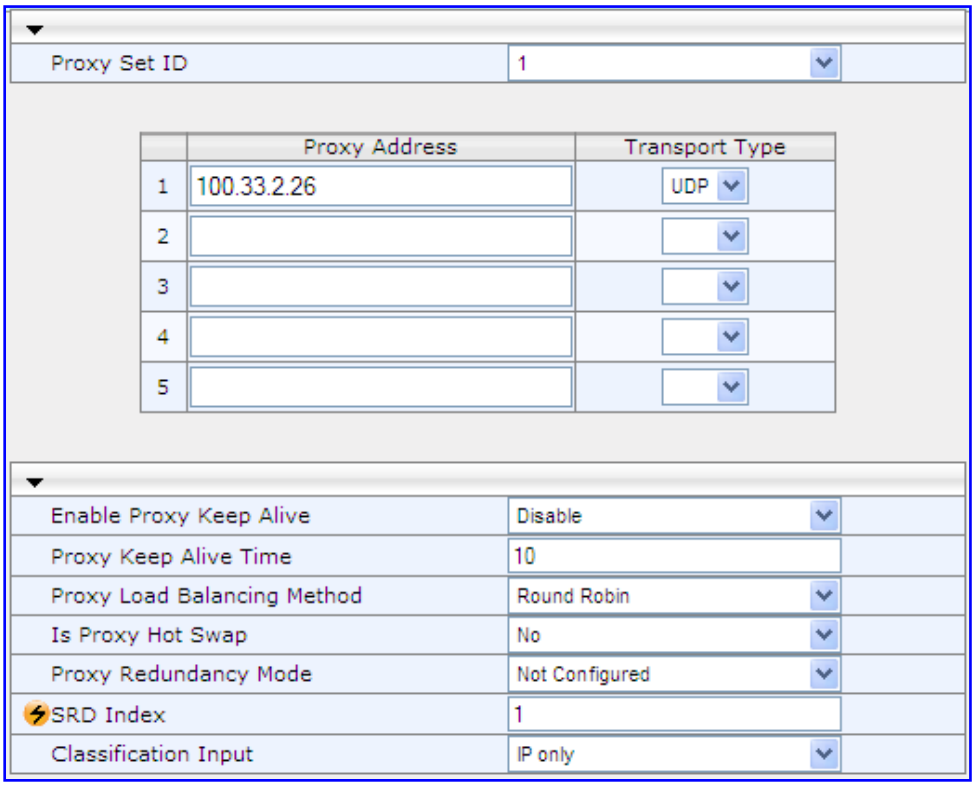
Proxy Sets can be assigned only to SERVER-type IP Groups.

Each IP Group can be classified according to its Proxy Set ID, if in the IP Group table the parameter ClassifyByProxySet is enabled.

To add Proxy servers:

1. Open the Proxy Sets Table page (**Configuration** tab >**VoIP** menu >**Control Network** submenu >**Proxy Sets Table**).

Figure 15-4: Proxy Sets Table Page



From the 'Proxy Set ID' drop-down list, select an ID for the desired group.

Configure the Proxy parameters according to the following table.

Click Subm**it to** apply your changes.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

Table 15-4: Proxy Sets Table Parameters

Parameter	Description
Web: Proxy Set ID EMS: Index **[ProxySet_Index]**	The Proxy Set identification number. The valid range is 0 to 31. The Proxy Set ID 0 is used as the default Proxy Set. Typically, when IP Groups are used, there is no need to use the default Proxy, and all routing and registration rules can be configured using IP Groups and the Account tables (see 'Configuring Account Table' on page 132).

Parameter	Description
Proxy Address **[ProxyIp_IpAddress]**	The IP address (and optionally port number) of the Proxy server. Up to five IP addresses can be configured per Proxy Set. Enter the IP address as an FQDN or in dotted-decimal notation (e.g., 201.10.8.1). You can also specify the selected port in the format: <IP address>:<port>. If you enable Proxy Redundancy (by setting the parameter EnableProxyKeepAlive to 1 or 2), the device can operate with multiple Proxy servers. If there is no response from the first (*primary*) Proxy defined in the list, the device attempts to communicate with the other (*redundant*) Proxies in the list. When a redundant Proxy is located, the device either continues operating with it until the next failure occurs or reverts to the primary Proxy (refer to the parameter ProxyRedundancyMode). If none of the Proxy servers respond, the device goes over the list again. The device also provides real-time switching (Hot-Swap mode) between the primary and redundant proxies (refer to the parameter IsProxyHotSwap). If the first Proxy doesn't respond to the INVITE message, the same INVITE message is immediately sent to the next Proxy in the list. The same logic applies to REGISTER messages. **Notes:** If EnableProxyKeepAlive is set to 1 or 2, the device monitors the connection with the Proxies by using keep-alive messages (OPTIONS or REGISTER). To use Proxy Redundancy, you must specify one or more redundant Proxies. When a port number is specified (e.g., domain.com:5080), DNS NAPTR/SRV queries aren't performed, even if ProxyDNSQueryType is set to 1 or 2.
Transport Type **[ProxyIp_TransportType]**	The transport type per Proxy server. **[0]** UDP **[1]** TCP **[2]** TLS **[-1]** = Undefined **Note:** If no transport type is selected, the value of the global parameter SIPTransportType is used (see 'Configuring SIP General Parameters' on page 130).

Parameter	Description
Web/EMS: Enable Proxy Keep Alive **[ProxySet_EnableProxyKeepAlive]**	Determines whether Keep-Alive with the Proxy is enabled or disabled. This parameter is configured per Proxy Set. **[0]** Disable = Disable (default). **[1]** Using Options = Enables Keep-Alive with Proxy using SIP OPTIONS messages. **[2]** Using Register = Enables Keep-Alive with Proxy using SIP REGISTER messages. If set to 'Using Options', the SIP OPTIONS message is sent every user-defined interval (configured by the parameter ProxyKeepAliveTime). If set to 'Using Register', the SIP REGISTER message is sent every user-defined interval (configured by the RegistrationTime parameter for the GW/IP2IP application or by the SBCProxyRegistrationTime parameter for SBC application). Any response from the Proxy, either success (200 OK) or failure (4xx response) is considered as if the Proxy is communicating correctly. **Notes:** For Survivability mode for USER-type IP Groups, this parameter must be enabled (1 or 2). This parameter must be set to 'Using Options' when Proxy redundancy is used. When this parameter is set to 'Using Register', the homing redundancy mode is disabled. When the active proxy doesn't respond to INVITE messages sent by the device, the proxy is tagged as 'offline'. The behavior is similar to a Keep-Alive (OPTIONS or REGISTER) failure. If this parameter is enabled and the proxy uses the TCP/TLS transport type, you can enable CRLF Keep-Alive mechanism, using the UsePingPongKeepAlive parameter.
Web: Proxy Keep Alive Time EMS: Keep Alive Time **[ProxySet_ProxyKeepAliveTime]**	Defines the Proxy keep-alive time interval (in seconds) between Keep-Alive messages. This parameter is configured per Proxy Set. The valid range is 5 to 2,000,000. The default value is 60. **Note:** This parameter is applicable only if the parameter EnableProxyKeepAlive is set to 1 (OPTIONS). When the parameter EnableProxyKeepAlive is set to 2 (REGISTER), the time interval between Keep-Alive messages is determined by the parameter RegistrationTime for the GW/IP2IP application or by the SBCProxyRegistrationTime parameter for SBC application.

Parameter	Description
Web: Proxy Load Balancing Method EMS: Load Balancing Method **[ProxySet_ProxyLoadBalancingMethod]**	Enables the Proxy Load Balancing mechanism per Proxy Set ID. **[0]** Disable = Load Balancing is disabled (default) **[1]** Round Robin **[2]** Random Weights When the Round Robinalgorithm is used, a list of all possible Proxy IP addresses is compiled. This list includes all IP addresses per Proxy Set, after necessary DNS resolutions (including NAPTR and SRV, if configured). After this list is compiled, the Proxy Keep-Alive mechanism (according to parameters EnableProxyKeepAlive and ProxyKeepAliveTime) tags each entry as 'offline' or 'online'. Load balancing is only performed on Proxy servers that are tagged as 'online'. All outgoing messages are equally distributed across the list of IP addresses. The IP addresses list is refreshed according to ProxyIPListRefreshTime. If a change in the order of the entries in the list occurs, all load statistics are erased and balancing starts over again. When the Random Weights algorithm is used, the outgoing requests are not distributed equally among the Proxies. The weights are received from the DNS server by using SRV records. The device sends the requests in such a fashion that each Proxy receives a percentage of the requests according to its' assigned weight. A single FQDN should be configured as a Proxy IP address. The Random Weights Load Balancing is not used in the following scenarios: The Proxy Set includes more than one Proxy IP address. The only Proxy defined is an IP address and not an FQDN. SRV is not enabled (DNSQueryType). The SRV response includes several records with a different Priority value.
Web/EMS: Is Proxy Hot-Swap **[ProxySet_IsProxyHotSwap]**	Enables the Proxy Hot-Swap redundancy mode per Proxy Set. **[0]** No (default) **[1]** Yes If Proxy Hot-Swap is enabled, the SIP INVITE/REGISTER message is initially sent to the first Proxy/Registrar server. If there is no response from the first Proxy/Registrar server after a specific number of retransmissions (configured by the parameter HotSwapRtx), the message is resent to the next redundant Proxy/Registrar server.
Web/EMS: Redundancy Mode **[ProxySet_ProxyRedundancyMode]**	Determines whether the device switches back to the primary Proxy after using a redundant Proxy (per this Proxy Set). **[-1]** = Not configured – the “global” parameter ProxyRedundancyMode applies (default). **[0]** Parking = The device continues operating with a redundant (now active) Proxy until the next failure, after which it operates with the next redundant Proxy. **[1]** Homing = The device always attempts to operate with the primary Proxy server (i.e., switches back to the primary Proxy whenever it's available). **Notes:** To use the Proxy Redundancy mechanism, you need to enable the keep-alive with Proxy option, by setting the parameter EnableProxyKeepAlive to 1 or 2. If this parameter is configured, then the global parameter is ignored.
Web/EMS: SRD Index	The SRD (defined in Configuring SRD Table on page 110) associated

Parameter	Description
[ProxySet_ProxySet_SRD]	with the Proxy Set ID. **Notes:** For this parameter to take effect, a device reset is required. If no SRD is defined for this parameter, by default, SRD ID #0 is associated with the Proxy Set.
Web/EMS: Classification Input **[ClassificationInput]**	Classifies an IP call to a Proxy Set, based on either its IP address, or based on its IP address, port, and transport type: **[0]** Compare only IP = IP call classified to Proxy Set according to IP address only (default). **[1]** Compare IP, port and transport type = IP call classified to Proxy Set according to IP address, port, and transport type.

15.5 Configuring NAT Translation per IP Interface

The NAT Translation table defines network address translation (NAT)rules for translating source IP addresses per VoIP interface (SIP control and RTP media traffic) into NAT IP addresses (*public* or *global*). This allows, for example, the separation of VoIP traffic between different ISTP's, and topology hiding of internal IP addresses to the "public" network. Each IP interface (configured in the Multiple Interface table - InterfaceTable parameter) can be associated with a NAT rule in this table, translating the source IP address and port of the outgoing packet into the NAT address (IP address and port range).

The device's priority method for performing NAT is as follows (not relevant for the SBC application):

Uses an external STUN server (STUNServerPrimaryIP parameter) to assign a NAT address to all interfaces.

Uses the StaticNATIP parameter to define one NAT IP address for all interfaces.

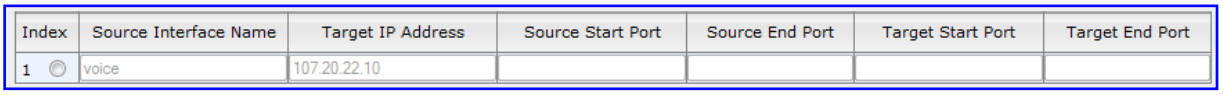
Uses the NATTranslation parameter to define NAT per interface.

If NAT is not configured (by any of the above-mentioned methods), the device sends the packet according to its IP address defined in the Multiple Interface table.

To configure NAT translation rules:

1. Open the NAT Translation Table page (**Configuration** tab >**VoIP** menu >**Control Network** submenu >**NAT Translation Table**).

Figure 15-5: NAT Translation Table Page



Configure the parameters according to the table below.

Click **Subm**ittoapply your changes.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

Table 15-5: NAT Translation Table Parameters

Parameter	Description
Index **[NATTranslation_Index]**	Defines the table index entry. This table can include up to 32 entries.
Source Interface Name **[NATTranslation_SourceIPInterfaceName]**	Defines the name of the IP interface, as appears in the Multiple Interface table.
Target IP Address **[NATTranslation_TargetIPAddress]**	Defines the global IP address.
Source Start Port **[NATTranslation_SourceStartPort]**	Defines the optional starting port range (1-65536) of the IP interface. If no ports are required, leave this field blank.
Source End Port **[NATTranslation_SourceEndPort]**	Defines the optional ending port range (1-65536) of the IP interface. If no ports are required, leave this field blank.
Target Start Port **[NATTranslation_TargetStartPort]**	Defines the optional starting port range (1-65536) of the global address. If no ports are required, leave this field blank.
Target End Port **[NATTranslation_TargetEndPort]**	Defines the optional ending port range (1-65536) of the global address. If no ports are required, leave this field blank.

15.6 Multiple SIP Signaling and Media Interfaces using SRDs

The device supports the configuration of multiple, logical SIP signaling interfaces and media (RTP) interfaces. Multiple SIP and media interfaces allow you to:

Separate SIP and media traffic between different applications (i.e., SAS and SBC)

Separate SIP and media traffic between different Layer-3 networks (e.g., when operating with multiple ITSPs - separation of signaling traffic between different customers). This separation allows you to use different routing rules, manipulations, SIP definitions, etc. per network (customer). This is also applicable for networks residing in the same or in different Layer-3 networks as the device. In such a scenario, the device is configured with multiple SRDs.

Due to the back-to-back user agent (B2BUA) nature of the SBC application, different interfaces can be assigned to each leg of the call.

Implement different SIP signaling ports (listening UDP, TCP, and TLS, and the UDP source ports) for single or multiple interfaces.

Only one signaling interface per application type is allowed per SRD. An SRD can be associated with many SIP interfaces which are based on one Layer-3 interface, with different ports.

Multiple SIP and RTP interfaces are implemented using SRDs (Signaling Routing Domains). An SRD is a set of definitions of IP interfaces, device resources, SIP behaviors and other definitions that together create (from the IP user's perspective), multiple, virtual multi-service gateways, from one physical device.

An SRD is composed of the following main entities:

Media Realm: A Media Realm is a range of UDP ports associated with a specific Media-type IP interface (defined in the Multiple Interface table in 'Configuring IP Interface Settings' on page 68). You can configure multiple Media Realms (each with a specified UDP port range) for a specific media IP interface, thereby allowing you to divide a media IP interface (RTP traffic) into a pool of media realms. Media Realms are configured in the Media Realm table (see 'Configuring Media Realms' on page 105). Once configured, you can assign Media Realms to an SRDs (and/or IP Groups).

SIP Interface: A SIP Interface is a combination of UDP, TCP, and/or TLS ports associated with a specific Control-type IP interface (defined in the Multiple Interface table). Therefore, a SIP Interface represents a SIP signaling interface. SIP Interfaces are configured n the SIP Interface table (see 'Configuring SIP Interface Table' on page 112) where they are assigned to SRDs:

- Each SIP Interface is defined with a unique signaling port (i.e., no two SIP Interfaces can share the same port - no overlapping).
- SIP Interfaces assigned to a specific SRD ID must all be defined with the same network interface (from the Multiple Interface table). For example, if you define three SIP Interfaces for SRD ID #8, all these SIP Interfaces must be defined with the same network interface (e.g., "SIP1").
- Each SIP Interface assigned to a specific SRD ID must be defined with a different application type (i.e., SAS and SBC). Therefore, up to three SIP Interfaces can be assigned to a specific SRD.

Once configured, you can use an SRD as follows:

Use it in classification rules for identifying incoming SIP requests (see Configuring Classification Table on page 178)

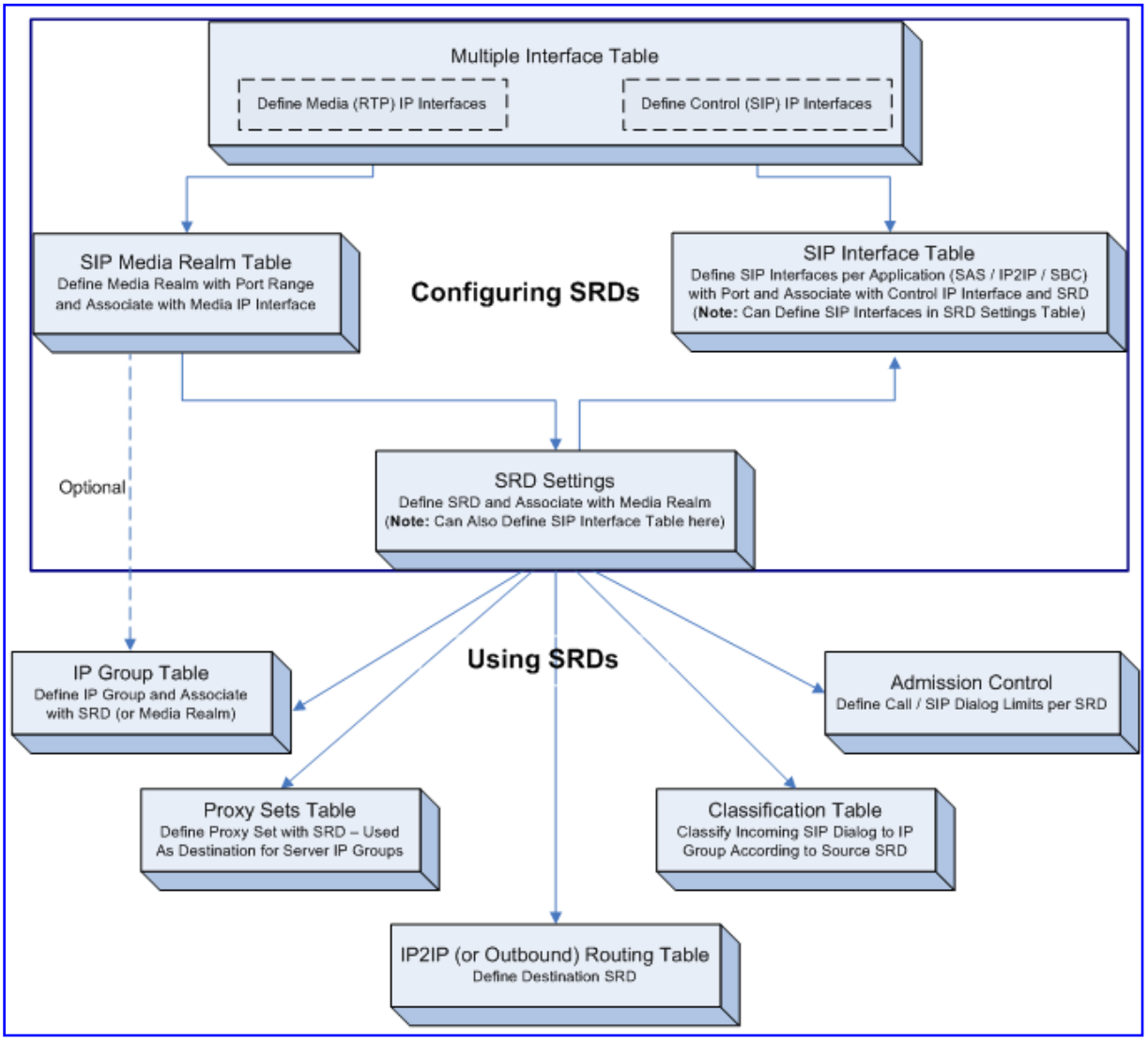
Associate it with an IP Group (see Configuring IP Groups on page 114).

Associate it with a Proxy Set (see Configuring Proxy Sets Table on page 120).

Define it as a destination SRD for IP-to-IP routing rules (see 'Configuring IP-to-IP Routing Table' on page 183). Routing from one SRD to another is possible, where each routing destination (IP Group or destination address) indicates the SRD to which it belongs.

Apply an Admission Control rule to it (see Configuring Admission Control Table on page 173).

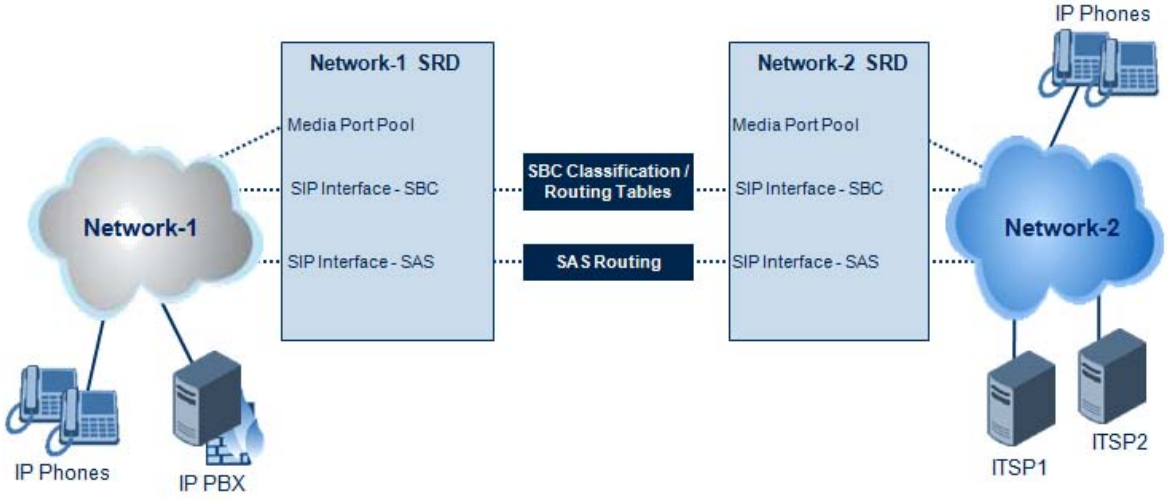
Figure 15-6: Configuring SRDs and Assignment


Multiple Interface Table
Define Media (RTP) IP Interfaces
Define Control (SIP) IP Interfaces
SIP Media Realm Table
Define Media Realm with Port Range and Associate with Media IP Interface
Configuring SRDs
SIP Interface Table
Define SIP Interfaces per Application (SAS / IP2IP / SBC) with Port and Associate with Control IP Interface and SRD (Note: Can Define SIP Interfaces in SRD Settings Table)
Optional
SRD Settings
Define SRD and Associate with Media Realm (Note: Can Also Define SIP Interface Table here)
Using SRDs
IP Group Table
Define IP Group and Associate with SRD (or Media Realm)
Admission Control
Define Call / SIP Dialog Limits per SRD
Proxy Sets Table
Define Proxy Set with SRD – Used As Destination for Server IP Groups
Classification Table
Classify Incoming SIP Dialog to IP Group According to Source SRD
IP2IP (or Outbound) Routing Table
Define Destination SRD

Typically, an SRD is defined per group of SIP UAs (e.g., proxies, IP phones, application servers, gateways, softswitches) that communicate with each other. This provides these entities with VoIP services that reside on the same Layer-3 network (must be able to communicate without traversing NAT devices and must not have overlapping IP addresses).

The figure below illustrates two SRD's - one for Network-1 and one for Network-2. Each application (i.e., SAS and SBC) pertains to the same SRD, but each has its own SIP interface.

Figure 15-7: SIP Interfaces per Application using Multiple SRDs (Example)



The figure below illustrates the SBC call flow between an enterprises LAN (IP PBX) and an ITSP (Network-2) implementing different interfaces (IP addresses and ports) for RTP packets and SIP signaling. In addition, for each leg, different interfaces are used.

The example uses the following IP addresses:

IP-PBX: 10.2.2.6

Network-1: 10.2.2.3

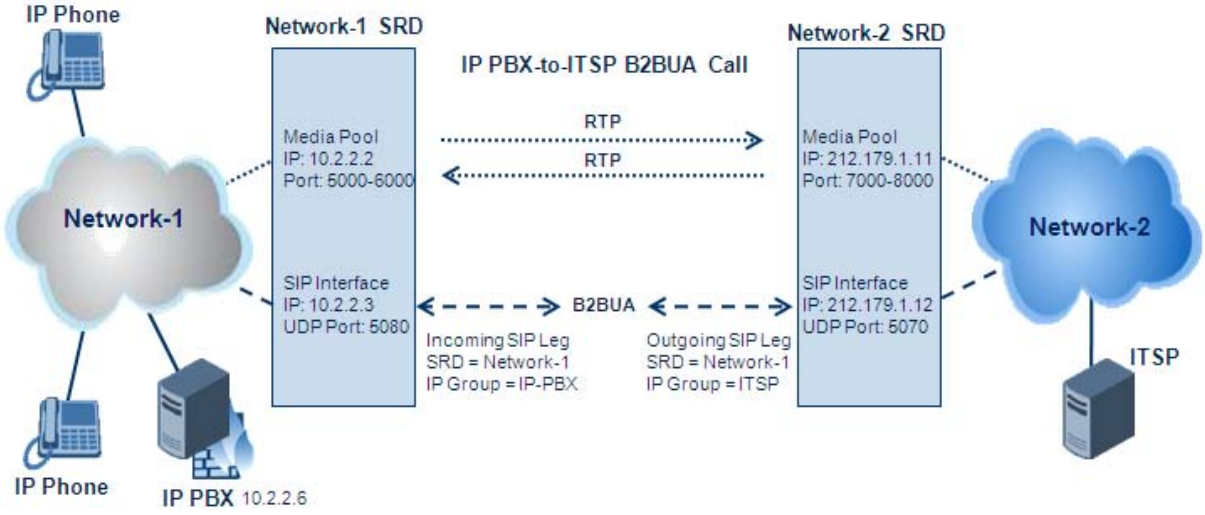
Network-2: 212.179.1.12

ITSP: 212.179.1.13

Network-1 Media: 10.2.2.2:5000-6000

Network-2 Media: 212.179.1.11:7000-8000

Figure 15-8: Back-to-Back SBC Call Flow (RTP and Signaling)



Reader's Notes

16 SIP Definitions

This section describes configuration of SIP parameters.

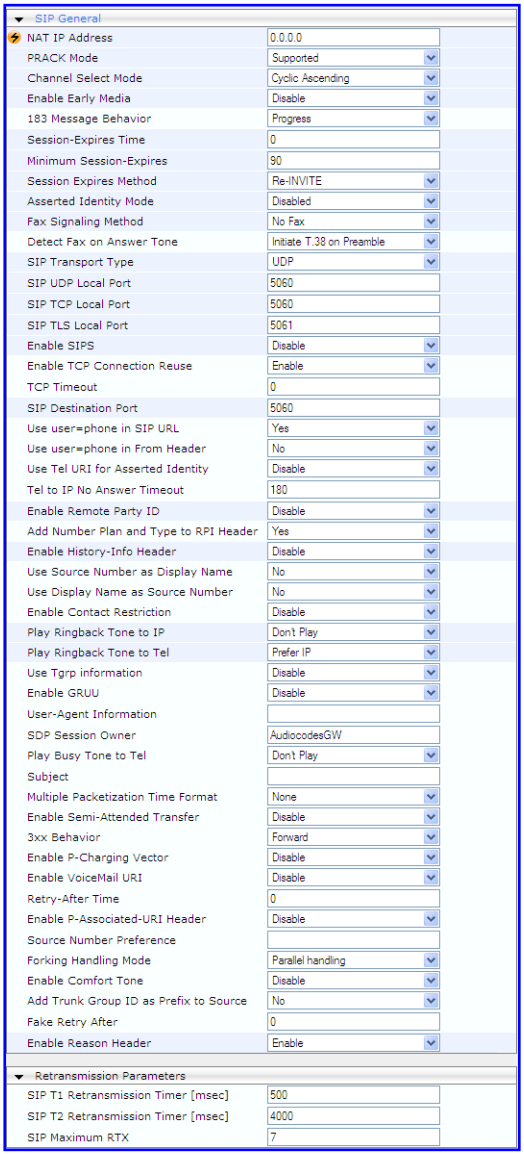
16.1 Configuring SIP General Parameters

The SIP General Parameters page is used to configure general SIP parameters. For a description of the parameters appearing on this page, see 'Configuration Parameters Reference' on page **Erreur ! Signet non défini.**.

To configure general SIP parameters:

1. Open the SIP General Parameters page (**Configuration** tab >**VoIP** menu >**SIP Definitions** submenu >**General Parameters**).

Figure 16-1: SIP General Parameters


SIP General
NAT IP Address 0.0.0.0
PRACK Mode Supported
Channel Select Mode Cyclic Ascending
Enable Early Media Disable
183 Message Behavior Progress
Session-Expires Time 0
Minimum Session-Expires 90
Session Expires Method Re-INVITE
Asserted Identity Mode Disabled
Fax Signaling Method No Fax
Detect Fax on Answer Tone Initiate T.38 on Preamble
SIP Transport Type UDP
SIP UDP Local Port 5060
SIP TCP Local Port 5060
SIP TLS Local Port 5061
Enable SIPS Disable
Enable TCP Connection Reuse Enable
TCP Timeout 0
SIP Destination Port 5060
Use user=phone in SIP URL Yes
Use user=phone in From Header No
Use Tel URI for Asserted Identity Disable
Tel to IP No Answer Timeout 180
Enable Remote Party ID Disable
Add Number Plan and Type to RPI Header Yes
Enable History-Info Header Disable
Use Source Number as Display Name No
Use Display Name as Source Number No
Enable Contact Restriction Disable
Play Ringback Tone to IP Don't Play
Play Ringback Tone to Tel Prefer IP
Use Tgrp information Disable
Enable GRUU Disable
User-Agent Information
SDP Session Owner AudiocodesGW
Play Busy Tone to Tel Don't Play
Subject
Multiple Packetization Time Format None
Enable Semi-Attended Transfer Disable
3xx Behavior Forward
Enable P-Charging Vector Disable
Enable VoiceMail URI Disable
Retry-After Time 0
Enable P-Associated-URI Header Disable
Source Number Preference
Forking Handling Mode Parallel handling
Enable Comfort Tone Disable
Add Trunk Group ID as Prefix to Source No
Fake Retry After 0
Enable Reason Header Enable
Retransmission Parameters
SIP T1 Retransmission Timer [msec] 500
SIP T2 Retransmission Timer [msec] 4000
SIP Maximum RTX 7

Configure the parameters as required.

Click Subm**it to** apply your changes.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

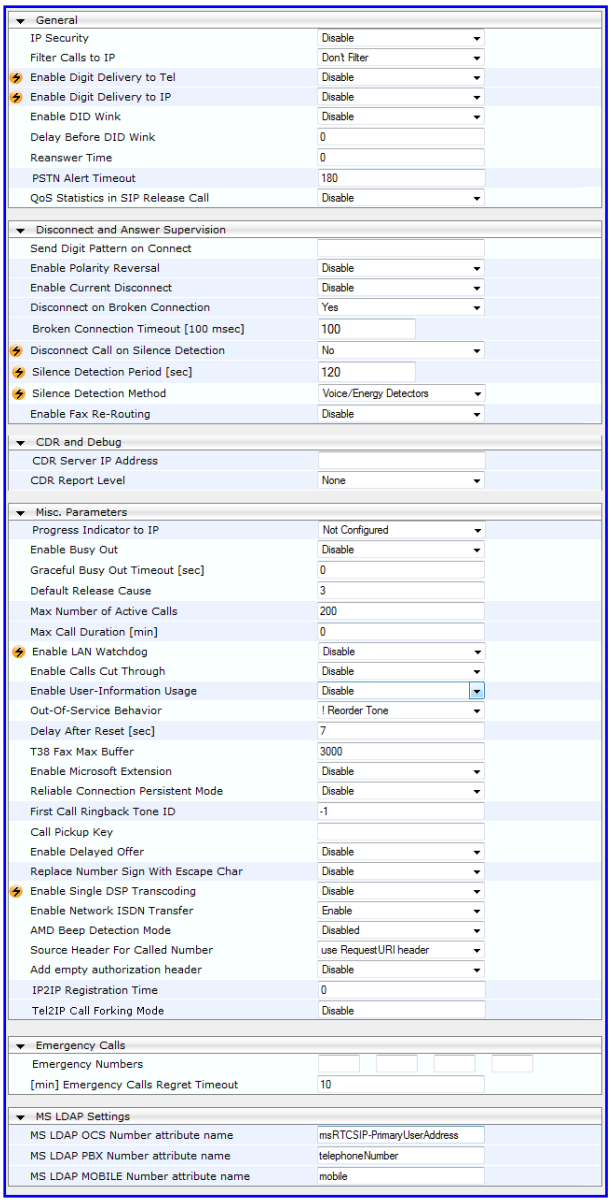
16.2 Configuring Advanced Parameters

The Advanced Parameters page allows you to configure advanced SIP control parameters. For a description of the parameters appearing on this page, see 'Configuration Parameters Reference' on page **Erreur ! Signet non défini.**.

To configure advanced general protocol parameters:

1. Open the Advanced Parameters page (**Configuration** tab >**VoIP** menu >**SIP Definitions** submenu >**Advanced Parameters**).

Figure 16-2: Advanced Parameters Parameters



Configure the parameters as required.

Click Subm**it to** apply your changes.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

16.3 Configuring Account Table

The Account Table page allows you to define up to 32*Accounts* per source IP Group (Served IP Group). This is used for registration and/or digest authentication (user name and password) to a destination IP address (*Serving IP Group*). The Account table can be used, for example, to register to an ITSP on behalf of an IP-PBX to which the device is connected. The registrations are sent to the Proxy Set ID (see 'Configuring Proxy Sets Table' on page 120) associated with these Serving IP Groups.

A source IP Group can register to more than one Serving IP Group (e.g., ITSP's). This can be achieved by configuring multiple entries in the Account table with the same Served IP Group, but with different Serving IP Groups, user name/password, host name, and contact user values.

Notes:

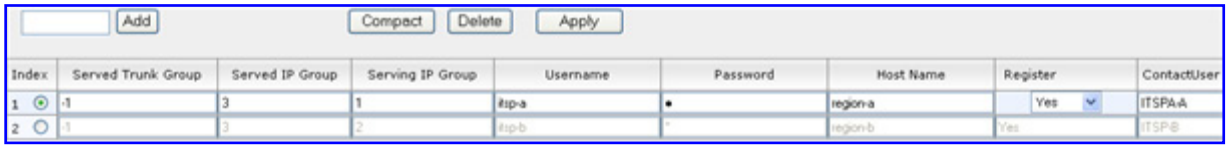
For viewing Account registration status, see 'Viewing Registration Status' on page 271.

You can also configure the Account table using the *ini* file table parameter Account (see 'Configuration Parameters Reference' on page **Erreur ! Signet non défini.**).

To configure Accounts:

1. Open the Account Table page (**Configuration** tab >**VoIP** menu >**SIP Definitions** submenu >**Account Table**).

Figure 16-3: Account Table Page



To add an Account, in the 'Add' field, enter the desired table row index, and then click **Add**. A new row appears.

Configure the Account parameters according to the table below.

Click the **Apply** button to save your changes.

To save the changes, see 'Saving Configuration' on page 245.

To perform registration, click the **Register** button; to unregister, click **Unregister**. .

Note: For a description of the Web interface's table command buttons (e.g., **Duplicate** and **Delete**), see 'Working with Tables' on page 28.

Table 16-1: Account Table Parameters Description

Parameter	Description
Serving IP Group **[Account_ServingIPGroup]**	The destination IP Group ID (defined in 'Configuring IP Groups' on page 114) to where the REGISTER requests (if enabled) are sent or authentication is performed. The actual destination to where the REGISTER requests are sent is the IP address defined for the Proxy Set ID (see 'Configuring Proxy Sets Table' on page 120) associated with the IP Group. This occurs only in the following conditions: The parameter 'Register' in this table is set to 1. In addition, for a SIP call that is identified by both the Served Served IP Group and Serving IP Group, the username and password for digest authentication defined in this table is used.
Username **[Account_Username]**	Digest MD5 Authentication user name (up to 50 characters).
Password **[Account_Password]**	Digest MD5 Authentication password (up to 50 characters). **Note:** After you click the **Apply** button, this password is displayed as an asterisk (*).
Host Name **[Account_HostName]**	Defines the Address of Record (AOR) host name. It appears in REGISTER From/To headers as ContactUser@HostName. For successful registrations, this HostName is also included in the INVITE request's From header URI. If not configured or if registration fails, the 'SIP Group Name' parameter from the 'IP Group' table is used instead. This parameter can be up to 49 characters.

Register
[Account_Register]
	Enables registration.
	[0] No = Don't register
	[1] Yes = Enables registration
	When enabled, the device sends REGISTER requests to the Serving IP Group. The Host Name (i.e., host name in SIP From/To headers) and Contact User (user in From/To and Contact headers) are taken from this table upon a successful registration. See the example below:
	`REGISTER sip:xyz SIP/2.0`
	`Via: SIP/2.0/UDP`
	`10.33.37.78;branch=z9hG4bKac1397582418`
	`From: <sip:ContactUser@HostName>;tag=1c1397576231`
	`To: <sip: ContactUser@HostName >`
	`Call-ID: 139756895726120002256@10.33.37.78`
	`CSeq: 1 REGISTER`
	`Contact:`
	`<sip:ContactUser@10.33.37.78>;expires=3600`
	`Expires: 3600`
	`User-Agent: Sip-Gateway/v.6.00A.008.002`
	`Content-Length: 0`

Contact User **[Account_ContactUser]**	Defines the AOR user name. It appears in REGISTER From/To headers as ContactUser@HostName, and in INVITE/200 OK Contact headers as ContactUser@<device's IP address>. If not configured, the 'Contact User' parameter in the IP Group Table page is used instead. **Note:** If registration fails, then the user part in the INVITE Contact header contains the source party number.
Application Type **[Account_ApplicationType]**	Defines the application type: **[2]** SBC = SBC application

16.4 Configuring Proxy and Registration Parameters

The Proxy & Registration page allows you to configure the Proxy server and registration parameters. For a description of the parameters appearing on this page, see 'Configuration Parameters Reference' on page **Erreur ! Signet non défini.**.

Note: To view whether the device or its endpoints have registered to a SIP Registrar/Proxy server, see 'Viewing Registration Status' on page 271.

To configure the Proxy and registration parameters:

1. Open the Proxy & Registration page (**Configuration** tab >**VoIP** menu >**SIP Definitions** submenu >**Proxy & Registration**).

Figure 16-4: Proxy & Registration Page

Configure the parameters as required.

Click Subm**it to** apply your changes.

Click the **Register** or **Un-Register** buttons to save your changes and register/unregister the device to a Proxy/Registrar.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

Click the **Proxy Set Table** button to Open the Proxy Sets Table page to configure groups of proxy addresses. Alternatively, you can open this page from the **Proxy Sets Table** page item (see 'Configuring Proxy Sets Table' on page 120for a description of this page).

17 Profiles

This section describes configuration of the SIP profiles parameters.

17.1 Configuring IP Profiles

The IP Profile Settings page allows you to define up to nine SIP profiles for IP calls (termed *IP Profile*). Each IP Profile contains a set of parameters for configuring various behaviors, for example, used coder, echo canceller support, and jitter buffer. Once configured, different IP Profiles can be assigned to specific inbound and outbound calls. For example, specific calls can be assigned an IP Profile that must use the G.711 coder. Thus, implementing IP Profiles provides high-level adaptation when connected to a variety of equipment and protocols (at both Tel and IP sides), each of which may require different system behavior.

The IP Profiles can be used in the following tables:

IP Group table - see 'Configuring IP Groups' on page 114

IP2IP Routing Table - see 'Configuring SBC IP-to-IP Routing' on page 183

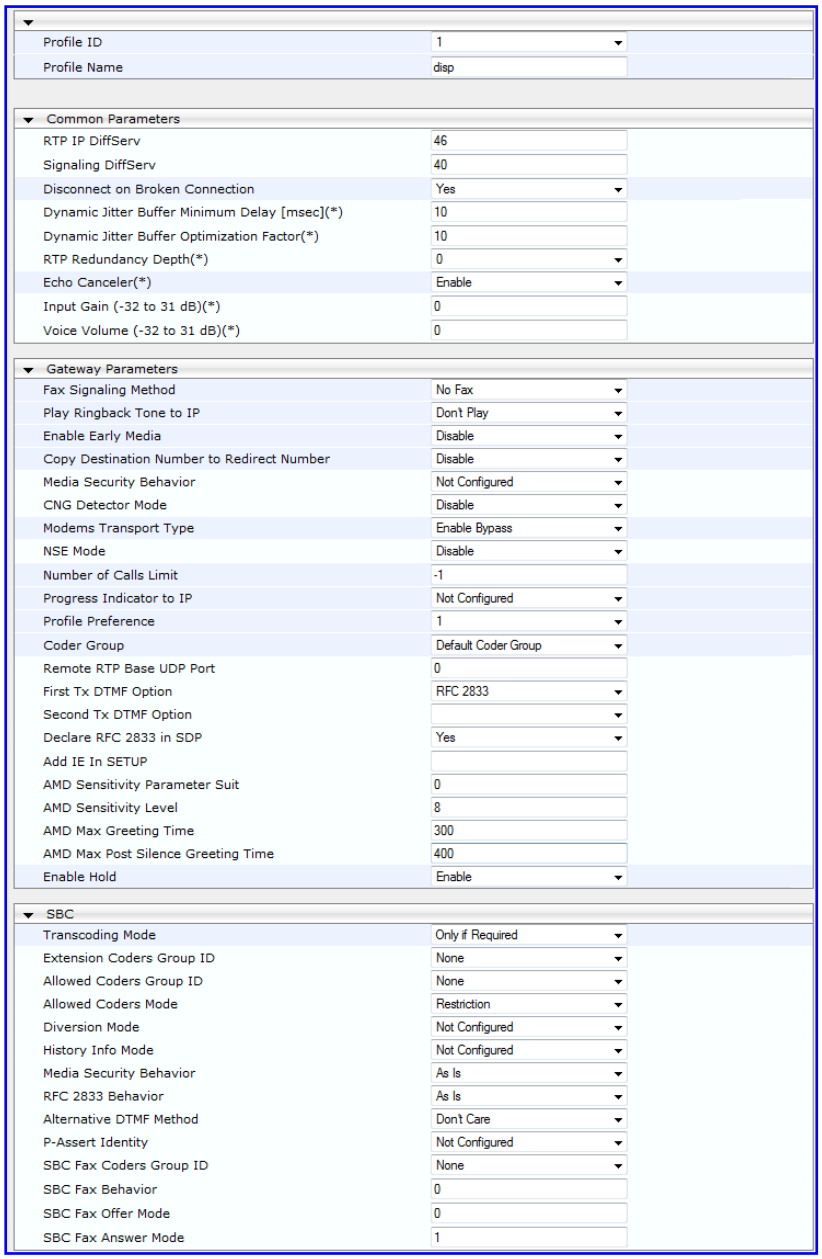
Notes:

- For a detailed description of each IP Profile parameter, refer to its corresponding "global" parameter (configured as an individual parameter).
- IP Profiles can also be implemented when operating with a Proxy server (when the AlwaysUseRouteTable parameter is set to 1).
- You can use IP Profiles in the IP Group table and IP2IP Routing table. The device selects the IP Profile as follows:
  1) If different IP Profiles (not default) are assigned to these tables, the device uses the IP Profile with the highest preference level (as set in the 'Profile Preference' field). If they have the same preference level, the device uses the IP Profile assigned to the IP Group table.
  2) If different IP Profiles are assigned to these tables and one table is set to the default IP Profile, the device uses the IP Profile that is not the default.
- You can also configure IP Profiles using the *ini* file table parameter IPProfile (see 'Configuration Parameters Reference' on page **Erreur ! Signet non défini.**).

To configure IP Profiles:

1. Open the IP Profile Settings page (**Configuration** tab >**VoIP** menu >**Coders And Profiles** submenu >**IP Profile Settings**).

Figure 17-1: IP Profile Settings Page



From the 'Profile ID' drop-down list, select the IP Profile index.

In the 'Profile Name' field, enter an arbitrary name that allows you to easily identify the IP Profile.

Configure the parameters as required.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

18 Session Border Controller

This section provides a detailed description of the device's SBC application.

Note: For guidelines on how to deploy your E-SBC device based on network topology and for SBC deployment examples, please refer to the *Typical SBC Deployments Guide*document.

18.1 SBC Overview

This section provides a detailed description of the device's SBC application.

This section includes the following subsections:

Overview of the SBC application (see 'Overview' on page 139)

SIP networking definitions (see 'SIP Network Definitions' on page 141)

SIP dialog-initiation process (see 'SIP Dialog Initiation Process' on page 141)

User registration and the device's database (see 'User Registration and Internal Database' on page 150)

Media handling (see 'SBC Media Handling' on page 154)

SBC Dialog Admission Control (see 'SIP Dialog Admission Control' on page 161)

Handling SIP 3xx Redirect Responses (see 'Handling SIP 3xx Redirect Responses' on page 162)

SIP Diversion and History-Info headers interworking (see 'Interworking SIP Diversion and History-Info Headers' on page 164)

18.1.1 Overview

The SBC application provides the following main features:

NAT traversal (see 'NAT Traversal' on page 140)

VoIP firewall and security for signaling and media (see 'VoIP Firewall' on page 140)

Topology hiding (see 'Topology Hiding' on page 140)

SIP normalization (see 'SIP Normalization' on page 141)

Survivability (see 'Survivability' on page 141)

Routing (see 'SIP Network Definitions' on page 141and 'SIP Dialog Initiation Process' on page 141):

- IP-to-IP routing translations of SIP, UDP, TCP, TLS (when extensive transcoding is not required)
- Load balancing and redundancy of SIP servers
- Routing according to Request-URI\Specific IP address\Proxy\FQDN
- Alternative routing
- Routing between different Layer-3 networks

Load balancing\redundancy of SIP servers

Internet Telephony Service Providers (ITSP) accounts

SIP URI user and host name manipulations (see 'SIP Dialog Initiation Process' on page 141)

18.1.1.1 NAT Traversal

The device supports NAT traversal, allowing, for example, communication with ITSPs with globally unique IP addresses, for LAN-to-WAN VoIP signaling (and bearer), using two independent legs. In addition, it also enables communication for "far-end" users located behind a NAT on the WAN. The device supports this by:

Continually registering far-end users in its dynamic database

Maintaining remote NAT binding state by frequent registrations, thereby, off-loading far-end registrations from the LAN IP PBX

Using Symmetric RTP (RFC 4961) to overcome bearer NAT traversal

18.1.1.2 VoIP Firewall

The device provides a firewall for VoIP:

SIP signaling:

- Deep and stateful inspection of all SIP signaling packets
- SIP dialog initiations may be rejected based on values of incoming SIP INVITE message and other Layer-3 characteristics
- Packets not belonging to an authorized SIP dialog are discarded

RTP:

- Opening pinholes (ports) in the device's firewall based on Offer-Answer SDP negotiations
- Deep packet inspection of all RTP packets
- Late rouge detection - if a SIP session was gracefully terminated and someone tries to "ride on it" with rouge traffic from the already terminated RTP and SIP context, the VoIP Firewall prevents this from occurring
- Disconnects call (after user-defined time) if RTP connection is broken
- Black/White lists for both Layer-3 firewall and SIP classification

18.1.1.3 Topology Hiding

The device intrinsically supports topology hiding, limiting the amount of topology information displayed to external parties. For example, IP addresses of ITSPs' equipment (e.g. proxies, gateways, and application servers) can be hidden from outside parties.

The device's topology hiding is provided by implementing back-to-back user agent (B2BUA) leg routing:

Strips all incoming SIP Via header fields and creates a new Via value for the outgoing message

Each leg has its own Route/Record Route set

Modifies SIP To, From, and Request-URI host names (must be configured using the Message Manipulations table - see 'Configuring Message Manipulations' on page 192)

Generates a new SIP Call-ID header value (different between legs)

Changes the SIP Contact header to the device's own address

Layer-3 topology hiding by modifying source IP address in the SIP IP header

18.1.1.4 SIP Normalization

The device supports SIP normalization, whereby the SBC application can overcome interoperability problems between SIP user agents. This is achieved by the following:

Manipulation of SIP URI user and host parts

Connection to ITSP SIP trunks on behalf of an IP-PBX - the device can register and utilize user and password to authenticate for the IP-PBX

18.1.1.5 Survivability

The device's SBC application provides two survivability features:

Routing calls to alternative routes

Routing calls between user agents in the local network using a dynamic database (built according to registrations of SIP user agents)

18.1.2 SIP Network Definitions

The device's SBC application can implement multiple SIP signaling and RTP (media) interfaces. For more information, see 'Mutiple SIP Signaling/Media Interfaces Environment' on page 127.

18.1.3 SIP Dialog Initiation Process

The device's SIP dialog initiation process concerns all incoming SIP dialog initiation requests. This includes SIP methods such as INVITE, SUBSCRIBE, OPTIONS, REFER, INFO, UNSOLICITED NOTIFY, MESSAGE, and REGISTER.

The SIP dialog initiation process consists of the following stages:

Determining Source and Destination URL (see 'Determining Source and Destination URL' on page 142)

Classifying Source IP Group (see 'Source IP Group Classification' on page 143)

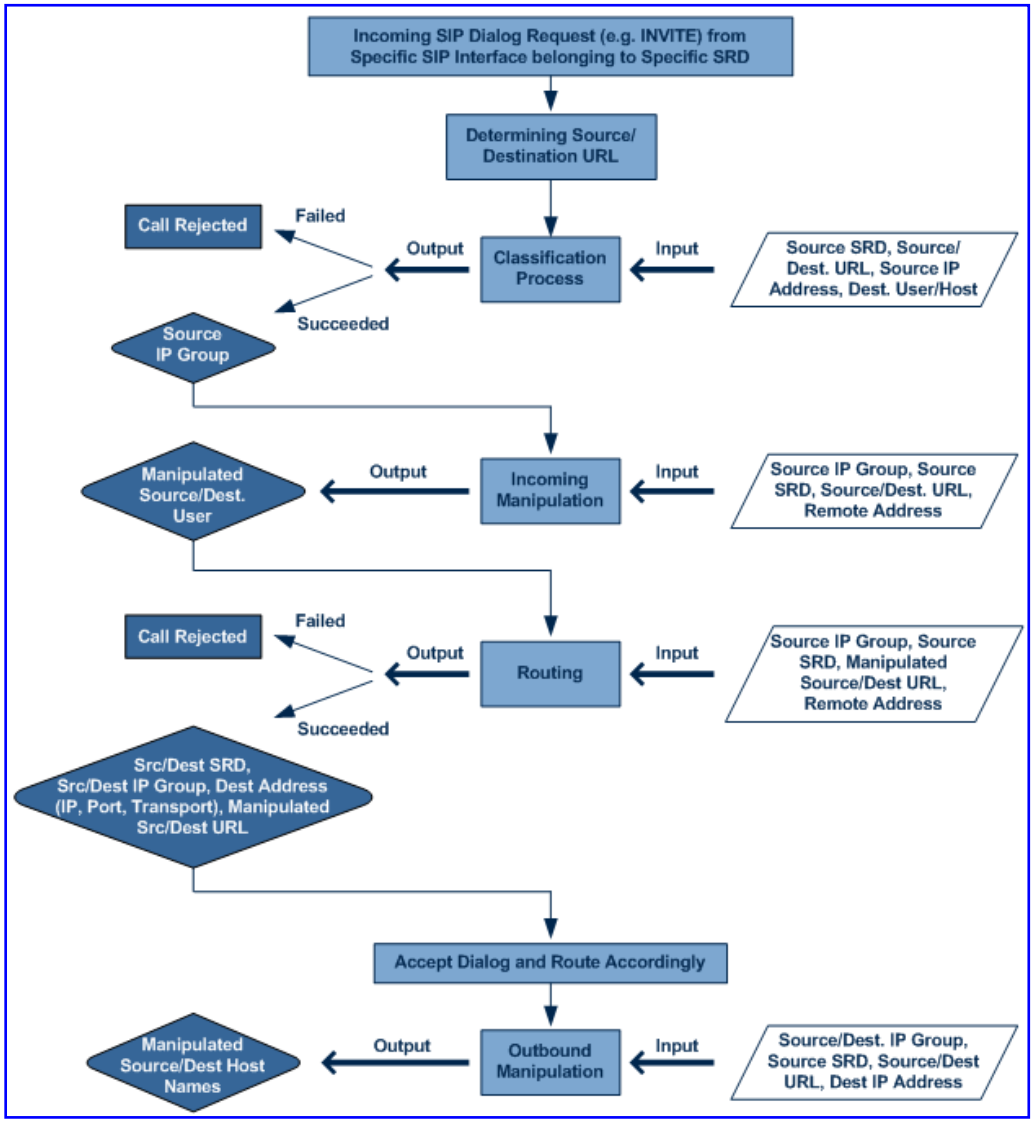
IP-to-IP Routing (see 'SBC IP-to-IP Routing' on page 145)

Manipulating IP-to-IP Inbound and Outbound SIP dialogs (see 'IP-to-IP Inbound and Outbound Manipulation' on page 145)

For a description of the Registration process, see 'User Registration and Internal Database' on page 150.

The flowchart below illustrates this process:

Figure 18-1: Routing Process



18.1.3.1 Determining Source and Destination URL

The SIP protocol has more than one URL in a dialog establishing request that might represent the source and destination URL. When handling an incoming request, the device determines which SIP headers are used for source and destination URLs. Once these URLs are determined, the input user and host are taken from them.

INVITE dialogs:

Source URL: if exists, obtained from the P-Asserted\Preferred-Identity header; otherwise, from the From header

Destination URL: obtained from the Request-URI

REGISTER dialogs:

Source URL: obtained from the To header

Destination URL: obtained from the Request-URI

18.1.3.2 Source IP Group Classification

The device supports the configuration of rules for classifying incoming SIP dialog initiating request. The classification identifies the incoming SIP dialog request as belonging to a specific IP Group (from where the SIP dialog request originated).

Classification begins with the device's Registration database, where it searches for a match by checking if the request arrived from a registered user:

Compares received Contact to the Contact of the registered user

Compares P-Asserted/From URL to the registered AOR

If the database search is unsuccessful, the classification process proceeds with locating a Proxy Set (associated with the SIP dialog request's IP address, or IP address, port, and transport type if the ClassificationInput parameter is enabled in the Proxy Set) and then finding a match with a corresponding IP Group in the IP Group table. Each IP Group can be classified according to its Proxy Set (if in the IP Group table the parameter ClassifyByProxySet is enabled). If enabled, the device classifies Requests arriving from the IP Group's Proxy Set as coming from this IP Group. The classification is done according to the Proxy IP list (in case of host names, then according to the dynamically resolved IP address list). Note that this classification is not relevant in cases where multiple IP Groups use the same Proxy Set.

If classification based on Proxy Set is unsuccessful, the device proceeds to the Classification table, which searches for a source IP Group based on the following matching rules: Source IP Address, Source Username Prefix, Source Host Prefix, Destination Username Prefix, Destination Host Prefix, and Source SRD.

If the above classification process fails to determine the source IP Group to which the incoming packet belongs, the call can either be rejected, or allowed and processed (by assigning it to the default IP Group of the default SRD). This last classification is determined by the parameter AllowUnclassifiedCalls.

This IP Group is afterwards used for the following purposes:

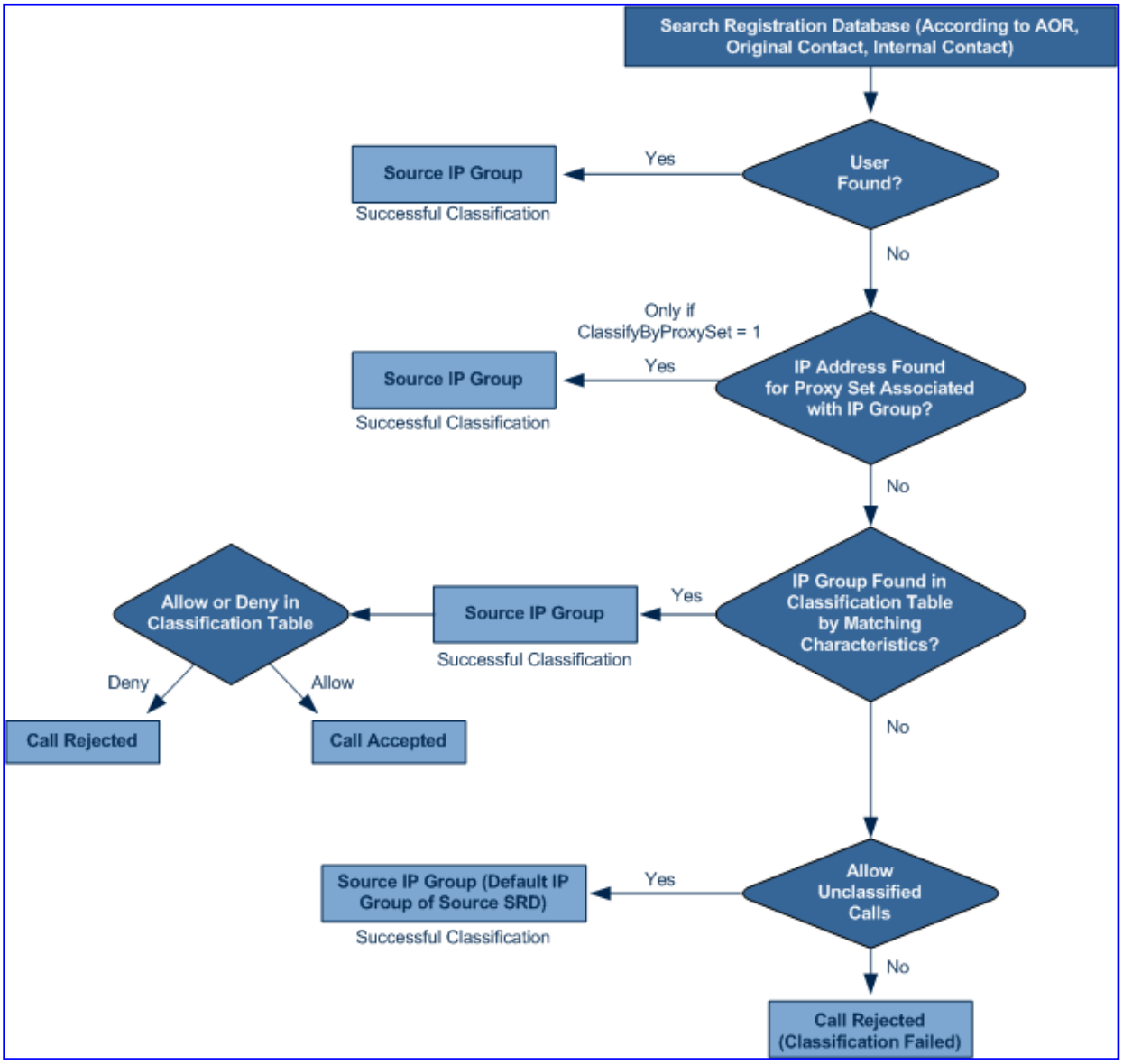
Input for the manipulation and routing processes

Defining SIP behavior and IP Profile, Media Realm and matching account

Note: Incoming REGISTER messages are recorded in the device's database and sent to a destination only if they are associated with a source IP Group that is of USER type.

The flowchart below illustrates the classification process:

Figure 18-2: Classification Process (Identifying IP Group or Rejecting Call)



18.1.3.3 SBC IP-to-IP Routing

The device's SBC application employs a comprehensive and flexible routing scheme:

Routing rules according to Layer-3/4 and SIP characteristics

Routing to different destination types:

- Request-URI (of incoming SIP dialog initiating requests)
- Specific destination IP address (based on IP address, host name, port, transport type, and/or SRD). Routing to a host name can be resolved using NAPTR/SRV/A-Record.
- Specific FQDN (NAPTR/SRV/A-Record Resolutions)
- Registered User Contact listed in the device's database (only for USER-type IP Groups)
- Destination IP Group (address defined by Proxy Set associated with the IP Group) with the ability of load balancing and redundancy
- ENUM query

Alternative Routing

Routing between two different Layer-3 networks

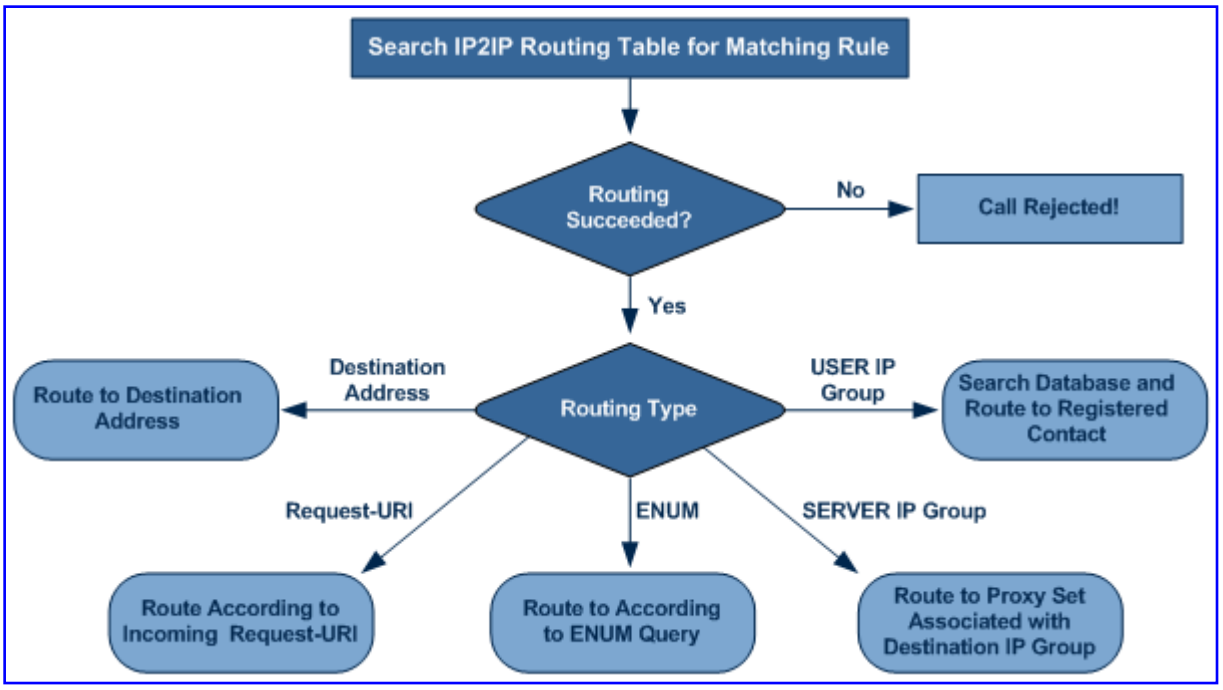
Transport protocol translator (UDP to TCP to TLS)

Source and destination user name manipulation (pre/post routing)

The device's IP-to-IP routing rules are configured in the IP-to-IP Routing table. This table provides enhanced IP-to-IP call routing capabilities for routing received SIP messages such as INVITE messages to a destination IP address. The routing rule must match one of the following input characteristics: Source IP Group, Source Phone Prefix, and/or Source Host Prefix.

For all destination types listed above except destination IP Group, the IP Group can optionally be itself, configured to provide destination SRD and/or IP Profile. If neither destination SRD nor destination IP Group is defined, the destination SRD is the source SRD and the destination IP Group is its default IP Group.

Figure 18-3: IP-to-IP Routing Types



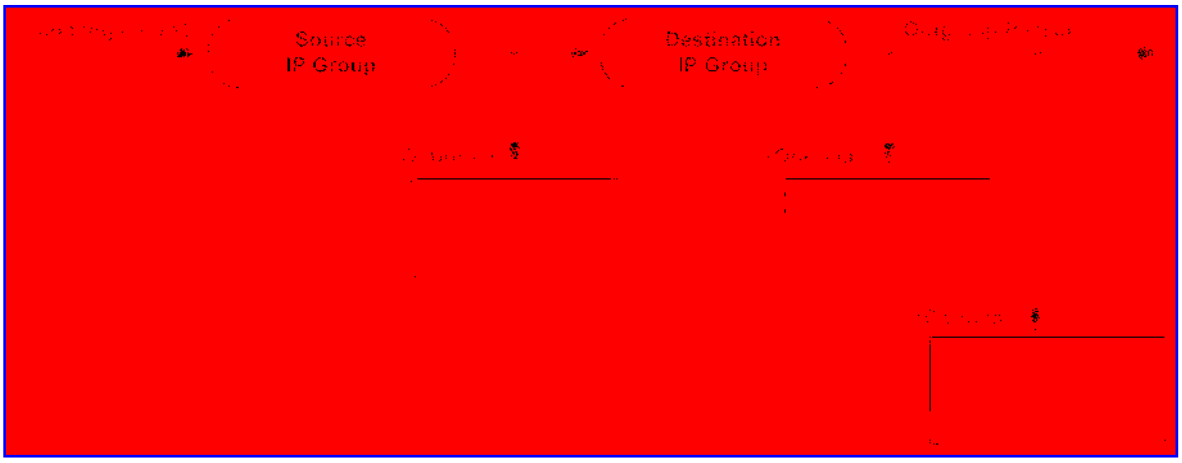
18.1.3.4 IP-to-IP Inbound and Outbound Manipulation

The device supports SIP URI user part (source and destination) manipulations for inbound and outbound routing. These manipulations can be applied to a source IP group, source and

destination host and user prefixes, and/or user-defined SIP request (e.g., INVITE, OPTIONS, SUBSCRIBE, and/or REGISTER). Since outbound manipulations are performed after routing, the outbound manipulation rule matching can also be done by destination IP Group.

Manipulated destination user and host are performed on the following SIP headers: Request-URI, To, and Remote-Party-ID (if exists). Manipulated source user and host are performed on the following SIP headers: From, P-Asserted (if exists), P-Preferred (if exists), and Remote-Party-ID (if exists).

Figure 18-4: SIP URI Manipulation in IP-to-IP Routing



In addition, you can restrict source user identity in outgoing SIP dialogs in the Outbound Manipulation table (using the column PrivacyRestrictionMode):

[0] Transparent (default): no device intervention in anything SIP data related to privacy

[1] Don't change privacy: the user identity remains the same as in the incoming INVITE. If a restricted number exists, the restricted presentation is normalized as follows:

From URL header - anonymous@anonymous.invalid.

If a P-Asserted-Identity header exists (either in the incoming INVITE or added by the device), a Privacy header is added with the value "id".

[2] Restrict: the user identity is restricted (the restricted presentation is as mentioned above).

[3] Remove Restriction: the device attempts to reveal the user identity by setting user values to the From header and removing the privacy "id" value if the Privacy header exists.

If the From header user is anonymous, the value is taken from the P-Preferred-Identity, P-Asserted-Identity, or Remote-Party-ID header (if exists).

The device identifies an incoming user as restricted if one of the following exists:

From header user is anonymous.

P-Asserted-Identity and Privacy headers contain the value "id".

All restriction logic is performed after the user number has been manipulated.

The manipulations are configured using the IPOutboundManipulation and IPInboundManipulation parameters.

Host name (source and destination) manipulations are simply host name substitutions with the names defined for the source and destination IP Groups respectively (if any, in the IP Group table).

Below is an example of a call flow and consequent SIP URI manipulations:

Figure 18-5: SIP INVITE (Manipulations)

Incoming INVITE (from Network-1)

```
INVITE sip:1000@10.2.2.3;user=phone;x=y;z=a SIP/2.0
Via: SIP/2.0/UDP 10.2.2.6;branch=z9hGLLLLLan
From:<sip:7000@10.2.2.6;user=phone;x=y;z=a>;tag=OILAN;paramer1=arik
To: <sip:1000@10.2.2.3;user=phone>
Call-ID: USELLLAN@10.2.2.3
CSeq: 1 INVITE
Contact: <sip:7000@10.2.2.3>
Supported: em,100rel,timer,replaces
Allow: REGISTER,OPTIONS,INVITE,ACK,CANCEL,BYE,NOTIFY,PRACK
User-Agent: Sip Message Generator V1.0.0.5
Content-Type: application/sdp
Content-Length: 155

v=0
o=SMG 791285 795617 IN IP4 10.2.2.6
s=Phone-Call
c=IN IP4 10.2.2.6
t=0 0
m=audio 6000 RTP/AVP 8
a=rtpmap:8 pcma/8000
a=sendrecv
a=ptime:20
```

Outgoing INVITE (to Network-2)

```
INVITE sip: 9721000@ITSP;user=phone;x=y;z=a SIP/2.0
Via: SIP/2.0/UDP 212.179.1.12;branch=z9hGWwan
From: <sip:97000@IP_PBX;user=phone;x=y;z=a>;tag=OWan;paramer1=arik
To: <sip: 9721000@ITSP;user=phone>
Call-ID: USEVWWAN@212.179.1.12
CSeq: 38 INVITE
Contact: <sip:7000@212.179.1.12>
Supported: em,100rel,timer,replaces
Allow: REGISTER,OPTIONS,INVITE,ACK,CANCEL,BYE,NOTIFY,PRACK,REFER
User-Agent: Sip Message Generator V1.0.0.5
Content-Type: application/sdp
Content-Length: 155

v=0
o=SMG 5 9 IN IP4 212.179.1.11
s=Phone-Call
c=IN IP4 212.179.1.11
t=0 0
m=audio 8000 RTP/AVP 8
a=rtpmap:8 pcma/8000
a=sendrecv
a=ptime:20
```

The SIP message manipulations in the example above (contributing to typical topology hiding) are as follows:

SIP Manipulation	From	To
Inbound Source SIP URI User Name	7000	97000 (blue)
Source IP Group Name (SIP URI Host Name)	10.2.2.6	IP_PBX (blue)
Inbound Destination SIP URI User Name	1000	9721000 (red)
Destination IP Group Name (SIP URI Host Name)	10.2.2.3	ITSP (red)

18.1.3.5 SIP Header Manipulation

The device provides enhanced SIP header manipulation, including insertion, removal, and/or modification of SIP headers and parameters. This manipulation is configured in the Message Manipulations table (MessageManipulations parameter). This feature enables the normalization of SIP messaging fields between communicating network segments. For example, it allows service providers to design their own policies on the SIP messaging fields that must be present before a SIP call enters their network. Similarly, enterprises and small businesses may have policies for the information that can enter or leave their networks for policy or security reasons from a service provider. The manipulations can also be implemented to resolve incompatibilities between SIP devices inside the enterprise network.

SIP Messaging manipulation supports the following:

Addition of new headers.

Removal of headers ("Black list").

Modification of header components - value, header value (e.g., URI value of the P-Asserted-Identity header can be copied to the From header), call's parameter values.

Deletion of SIP body (e.g., if a message body isn't supported at the destination network this body is removed).

Translating one SIP response code to another.

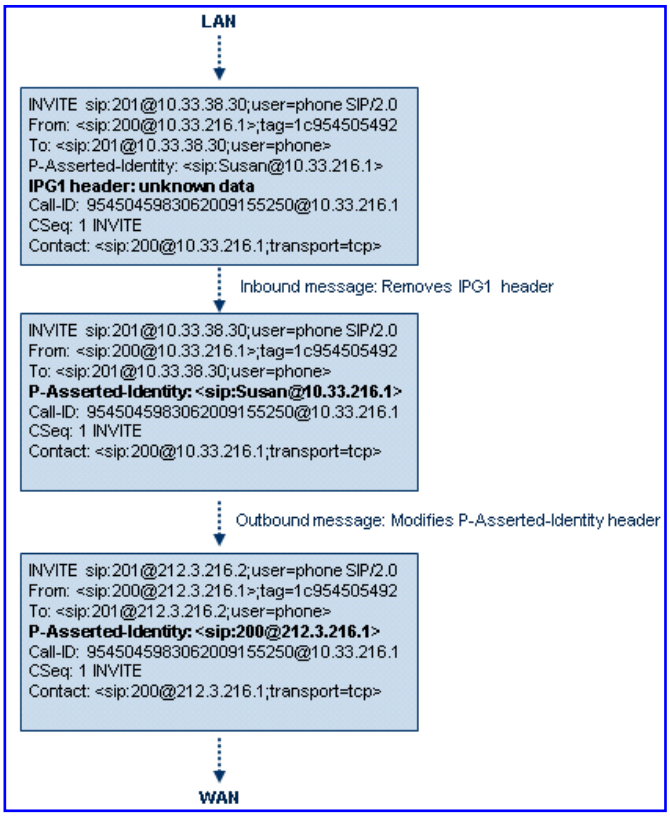
Topology hiding (generally present in SIP headers such as Via, Record Route, Route and Service-Route).

Configurable identity hiding (information related to identity of subscribers for example, P-Asserted-Identity, Referred-By, Identity and Identity-Info).

Apply conditions per rule - the condition can be on parts of the message or call's parameters.

Multiple manipulation rules on the same SIP message.

Figure 18-6: SIP Header Manipulation Example



The manipulation is performed on SIP messages according to the Classification table (source/destination of username/host prefixes and source IP address). The manipulation can be

performed on message type (Method, Request/Response, and Response type). Message manipulations are performed only after the classification, inbound manipulations and routing are successfully preformed (i.e., manipulations are performed only in the outgoing leg).

SIP Message manipulation rules can be assigned to an IP Group in the IP Group table (IPGroup parameter) and determined whether they must be performed for inbound or outbound messages.

Notes:

Unknown SIP parts can only be added or removed.

SIP manipulations do not allow you to remove or add mandatory SIP headers. Only the modify option is available for mandatory headers and is performed only on requests that initiate new dialogs. Mandatory SIP headers include To, From, Via, CSeq, Call-Id, and Max-Forwards. Mandatory SDP headers include v, o, s, t ,c, and m.

When multiple manipulations rules apply to the same header, the second rule applies to the result string of the first rule.

Manipulating any value in the message body causes a change to the Content-length header automatically.

SDP body manipulations are currently not supported.

18.1.4 User Registration and Internal Database

To allow registrations to traverse the SBC, the device must be configured with at least one IP Group of type USER. USER IP Groups represent a group of user agents that share the following characteristics:

Perform registrations and share the same serving proxy\registrar

Possess identical SIP and media behavior

Reside on the same Layer-3 network and are associated with the same SRD

Typically, the device is configured as the user agent's outbound proxy and the device is configured (using the IP2IP Routing table) to route requests received from this IP Group to the serving proxy and vice versa. Survivability can be achieved using the alternative routing feature.

18.1.4.1 Initial Registration Request Processing

Registration requests have different processing policies than other SIP methods:

1. Determining source and destination URL's:

   The source URL is obtained from the To header

   The destination URL is obtained from the Request URI

Classification: The REGISTER classification process is the same as the general classification process (described in previous sections). The source IP Group must be of type USER. If classification fails or the source IP Group is not of type USER, the registration is rejected.

Routing: The REGISTER routing is performed using the IP2IP Routing table:

The destination type can be an IP Group, specific IP address, Request-URI, or ENUM query (can also use DNS queries).

If the destination IP Group is of type USER, then the registration is not be forwarded. Instead, the device accepts (replies with 200 OK response) or rejects (Reply with 4xx) the request according to the user group policy.

Internal registration database: If the source IP Group is of type User and registration succeeds (replied with 200 OK by the IP-PBX), then the device adds a record to its database that identified the specific contact of this specific user (AOR). This record is used later to route requests to this specific user (either in normal or in survivability modes).

Alternative Routing: Alternative routing can be configured in the IP2IP Routing table for REGISTER requests.

Inbound Manipulation: The SBC record in the device's database includes the Contact header. Every REGISTER request is added to the database before manipulation, allowing correct user identification in the SBC Classification process for the next received request.

Session Admission Control: Applies various limitations on incoming and outgoing REGISTER requests. For example, limiting REGISTER requests from a certain IP Group/SRD. Note that this limitation is only for concurrent register dialogs and not concurrent registrations in the internal database.

The device can retain the original value of the SIP Expires header received from the user or proxy, in the outgoing REGISTER message. This feature also applies when the device is in “survivability” state (i.e., REGISTER requests cannot be forwarded to the proxy and is terminated by the device). This is configured by the SBCUserRegistrationTime, SBCProxyRegistrationTime, and SBCSurvivabilityRegistrationTime parameters.

By default, the Contact of the outgoing REGISTER is populated with a unique Contact generated by the device and associated with this specific registration. Alternatively, the original user can be retained in the Contact and used in the outgoing REGISTER request (using the SBCKeepContactUserinRegister parameter).

18.1.4.2 Internal Database

The device manages a dynamic database that is updated according to registration requests that traverse the SBC. Each database entry represents a binding between an AOR and one or more contact. Database bindings are added upon successful registration responses. For specific registrations, the AOR is obtained from the SIP To header and the contact is taken from the SIP Contact header.

Database bindings are removed in the following cases:

Successful de-registration responses (REGISTER with Expires header that equals zero)

Registration failure responses

Timeout of the Expires header value (in scenarios where the user agent did not send a refresh registration request)

The device's database can include up to 600 registered SBC users.

The database has the following limitations:

Maximum of five contacts per AOR

The same contact cannot belong to more than one AOR

Contacts with identical URIs and different ports and transport types are not supported (same key is created)

Multiple contacts in a single REGISTER is not supported

One database is shared between all USER-type IP Groups

18.1.4.3 Routing using Internal Database

Typically, routing using the database is applicable to all method types other than registrations. To route to a registered user (using the internal dynamic database), the following steps must be taken:

1. An IP2IP Routing rule with the desired input parameters (matching characteristics) and the destination type as IP Group (operation rule).

The destination IP Group must be of type USER.

To find a match for these specific rules, the device attempts to locate a match between the incoming Request-URI and (according to the description order):

a. Unique contact - the Contact generated by the SBC and sent in the initial registration request to the serving proxy

Registered AOR - the AOR of the incoming REGISTER request

Registered contact - the Contact of the incoming REGISTER request

If registrations are destined to the database (using the above rules), the device does not attempt to find a database match, but instead replies with 200 OK (used for Survivability). Once a match is found, the request is routed either to the contact received in the initial registration or (if the device identifies that the user agent is behind a NAT) to the source IP address of the initial registration.

18.1.4.4 Registration Refreshes

Registration refreshes are incoming REGISTER requests that are associated with a specific registered user. The association is performed by searching the internal registration database. These refreshes are routed to the serving proxy only if the serving proxy Expires time is about to expire; otherwise, the device responds with a 200 OK without routing the REGISTER. Each such refreshes also refresh the internal timer time set on the device for this specific registration.

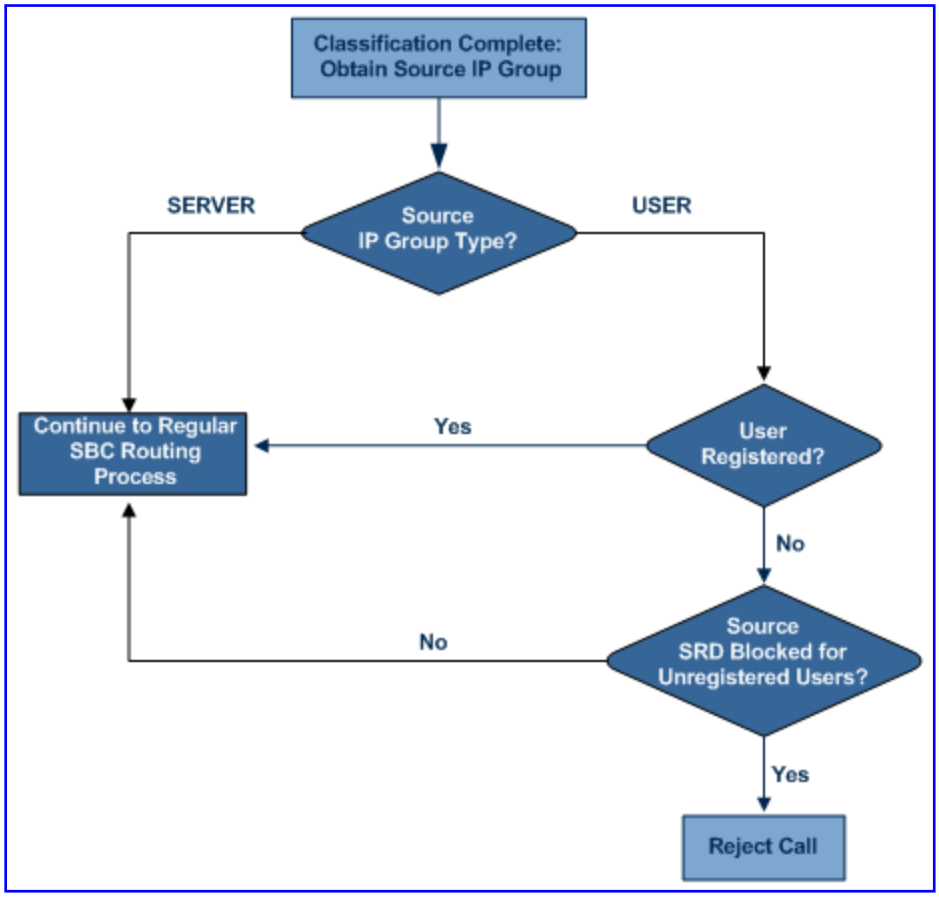
18.1.4.5 Registration Restriction Control

The device provides flexibility in controlling user's registration:

Limiting Number of Registrations per Source SRD and/or IP Group: You can limit the number of users that can register with the device. This limitation can be applied per source IP Group and/or SRD. By default, no limitation exists for registered users. This is configured using the parameters SRD or IPGroup.

Blocking Incoming Calls from Unregistered Users: You can block incoming calls (INVITE requests) from unregistered users (pertaining to USER-type IP Groups). By default, calls from unregistered users are not blocked. This is configured using the parameter SRD. The flowchart below depicts the process for blocking unregistered users. When the call is rejected, the device sends a SIP 500 "Server Internal Error" response to the remote end.

Figure 18-7: Blocking Incoming Calls from Unregistered Users



18.1.5 SBC Media Handling

Media behavior includes anything related to the establishment, management and termination of media sessions within the SIP protocol. Media sessions are created using the SIP "offer"/"answer" mechanism. If successful, the result is a bi-directional media (RTP) flow (e.g. audio, fax, modem, DTMF). Each offer/answer may create more than one media session of different types (e.g. audio and fax). In a SIP dialog, multiple offer/answer transactions may occur, each may change the media sessions characteristics (e.g. IP address, port, coders, media types, and RTP mode). The media capabilities exchanged in an offer/answer transaction include the following:

Media types (Audio, Secure Audio, Video, Fax, Text...)

IP addresses and ports of the media flow

Media flow mode (send receive, receive only, send only, inactive)

Media coders (coders and their characteristics used in each media flow)

Other (standard or proprietary) media and session characteristics

Even though the device usually does not change the negotiated media capabilities (mainly performed by the remote user agents), it does examine the media exchange to control negotiated media types (if necessary) and to know how to open the RTP media channels (IP addresses, coder type, payload type etc.).

The device is aware and sometimes active in the offer\answer process due to the following:

NAT traversal: the device changes the SDP address to be its own address, thereby, resolving NAT problems.

Firewall and security:

- RTP pin holes - only RTP packets related to a successful offer\answer negotiation traverse the device: When the device initializes, there are no RTP pin holes opened, this means that each RTP\RTCP packets destined to the device are discarded. Once an offer\answer transaction ends successfully, an RTP pin hole is opened and RTP\RTCP flows between the two remote user agents. Once a pin hole is opened, the payload type and RTP header version is validated for each packet. RTP pin holes close if one of the associated SIP dialogs is closed (may also be due to broken connection).
- Late rogue detection - once a dialog is disconnected, the related pin holes also disconnect.
- Deep Packet inspection of the RTP that flows through the opened pin holes.

Adding of media functionality to SIP user agents:

- Transcoding (for a description on the transcoding modes, see 'Transcoding Modes' on page 157)
- Broken connection

According to the above functionalities, the call can be configured to operate in one of the following modes:

Media Anchoring without Transcoding (Transparent): RTP traverses the device with minimal RTP packet changes (no DSP resources needed). This is typically used to solve NAT, firewall, and security issues. In this mode, all the "audio" coders in the received offer are included in the SBC outgoing offer. The Coder Table configuration has no effect on the coders in the outgoing offer. For more information, see 'Media Anchoring without Transcoding (Transparent)' on page 155.

No Media Anchoring: The RTP packet flow does not traverse the device. Instead, the two SIP UA's establish a direct RTP/SRTP flow between one another (see 'No Media Anchoring' on page 156).

18.1.5.1 Media Anchoring without Transcoding (Transparent)

To direct the RTP to flow through the device (for NAT traversal, firewall and security), all IP address fields in the SDP are modified:

Origin: IP address, session and version id

Session connection attribute ('c=' field)

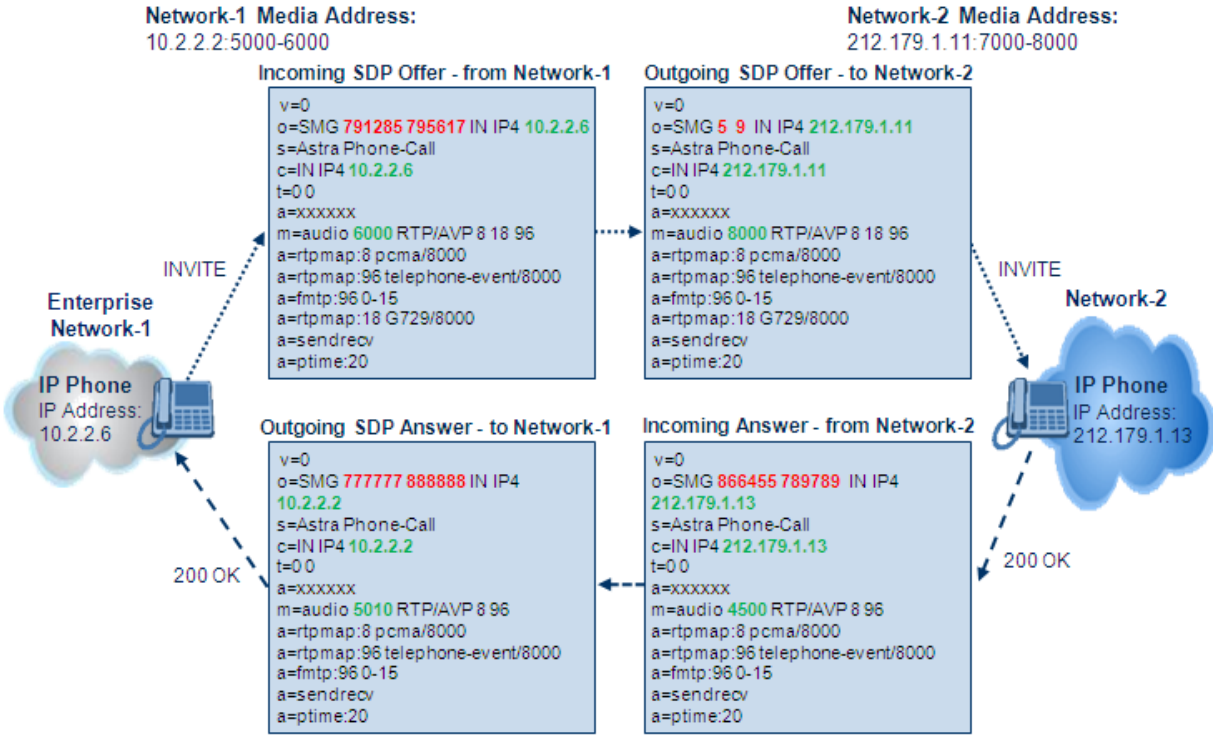
Media connection attribute ('c=' field)

Media port number

RTCP media attribute IP address and port (if the parameter EnableRTCPAttribute is set to 1)

Each SBC leg allocates and uses the device's local ports (e.g., for RTP\RTCP\fax). The local ports are allocated from a Media Realm associated with each leg. The legs are associated with a Media Realm as follows: If the leg's IP Group is configured with a Media Realm, then this is the associated Media Realm; otherwise, the leg's SRD Media Realm is the associated one. The figure below illustrates an example of SDP handling for a call between IP Phone 10.2.2.6 (Network-1) and a remote IP Phone 212.179.1.13 (Network-2).

Figure 18-8: SDP Offer/Answer Example



18.1.5.2 No Media Anchoring

The No Media Anchoring feature enables the use of SBC signaling capabilities without handling the RTP/SRTP (media) flow between remote SIP user agents (UA). The RTP packet flow does not traverse the device, instead, the two SIP UA's establish a direct RTP/SRTP flow (i.e., direct call) between one another. Signaling continues to traverse the device with minimal intermediation and involvement to enable certain SBC abilities such as routing.

In contrast to the regular SBC implementation, the No Media Anchoring feature:

Does not perform any manipulation on SDP data (offer/answer transaction) such as ports, IP address, coders.

Opening voice channels and allocation of IP media ports are not required.

The No Media Anchoring feature is typically implemented in the following scenarios:

SBC device is located within the LAN.

Calls between two SIP UA's in the same LAN and signals are sent to a SIP proxy server that is located in a different network.

The benefits of implementing the No Media Anchoring feature include the following:

Saves network bandwidth

Reduces CPU usage (no RTP/SRTP handling)

Avoids interference in SDP negotiation and header manipulation on RTP/SRTP

The No Media Anchoring process is as follows:

1. Identifies a No Media Anchoring call - according to configuration and the call's properties (such as source, destination, IP Group, and SRD).

Handles the identified No Media Anchoring call.

The No Media Anchoring feature is enabled using the SBCDirectMedia parameter. You can also enable No Media Anchoring per SRD (using the IntraSRDMediaAnchoring parameter), whereby calls between two UA's that pertain to the same SRD (source and destination) are handled as No Media Anchoring (direct media) calls.

Notes:

No Media Anchoring can be used when the SBC does not do NAT traversal (for media) where all the users are in the same domain.

No Media Anchoring calls cannot operate simultaneously with the following SBC features:
- Extension of RFC 2833/Out-of-band DTMF/In-band DTMF
- Extension of SRTP/RTP

All restriction features (Allowed Coders, restrict SRTP/SRT, restrict RFC 2833) can operate simultaneously. Once No Media Anchoring is enabled, the features listed above are disabled.

The Coder Restriction feature operates simultaneously with No Media Anchoring calls. Restricted coders are removed from the SDP offer message.

When two UA's pertain to the same SRD, the parameter IntraSRDMediaAnchoring is set to 1, and one of the UA's is defined as a foreign user (example, "follow me service") located in the WAN, while the other UA is located in the LAN: calls between these two UA's can't be established until IntraSRDMediaAnchoring is set to 0, as the device doesn't interfere in the SIP signaling. In other words, parameters such as IP addresses are not manipulated for calls between LAN and WAN (although required).

When the parameter SBCDirectMedia is disabled, No Media Anchoring calls between two UA's belonging to separate SRD's cannot be configured. No Media Anchoring calls between two UA's belonging to the same SRD is configurable only (in this case).

18.1.5.3 Interworking DTMF Methods

The device supports interworking between various DTMF methods such as RFC 2833, In-Band DTMF's, and SIP INFO (Cisco\Nortel\Korea). By default, the device allows the remote user agents to negotiate (in case of RFC 2833) and passes DTMF without intervention. However, if two user agents (UA) support different DTMF methods, the device can interwork these different DTMF methods at each leg.

This DTMF interworking feature is enabled using IP Profiles (*ini* file parameter IPProfile):

SBCRFC2833Behavior - affects the RFC 2833 SDP offer\answer negotiation:

[0] (default): the device does not intervene in the RFC 2833 negotiation.

[1]: each outgoing offer\answer includes RFC 2833 in the offered SDP (the device adds RFC 2833 only if the incoming offer does not include RFC 2833).

[2]: the device removes RFC 2833 from the incoming offer.

SBCAlternativeDTMFMethod – the device's first priority for DTMF method at each leg is RFC 2833. Therefore, if a specific leg negotiates RFC 2833 successfully, then the chosen DTMF method for this leg is RFC 2833. For legs where RFC 2833 is not negotiated successfully, the device uses this parameter to determine the DTMF method for the leg.

[0] (default): the device does not attempt to interwork any special DTMF method

[1]: In Band

[2]: INFO, Cisco

[3]: INFO, Nortel

[4]: INFO, Korea

The chosen DTMF method determines (for each leg) which DTMF method is used for sending DTMF's. If the device interworks between different DTMF methods and one of the methods is In-band\RFC 2833, detection and generation of DTMF methods requires DSP allocation.

18.1.5.4 Restricting Coders

The SBC Allowed Coders (coders restriction) feature determines the coders that can be used for a specific SBC leg. This provides greater control over bandwidth by enforcing the use of specific coders (*allowed coders groups*) while preventing the use of other coders. This is done by defining a group of allowed coders for the SBC leg, as described below:

1. Configure a Coders Group for allowed coders, using the AllowedCodersGroup parameter.

Select this Coders Group using the SBCAllowedCodersGroupID parameter of the IP Profile table.

Enable this feature by setting the SBCAllowedCodersMode parameter of the IP Profile table to **Restriction** or **Restriction and Preference**.

Coders that are not listed in the Allowed Coders Group are removed from the SDP offer. Therefore, only coders common between the SDP offer and Allowed Coders Group are used.

In addition to restricting the use of coders, the device can prioritize the coders listed in the SDP offer. This feature is referred to as *Coder Preference*. This is done on both SBC legs:

Incoming SDP offer: The coders list is re-arranged according to their order in the Allowed Coders Group table. The coders listed higher up in the table take preference over ones listed lower down in the table. This feature is enabled by setting the SBCAllowedCodersMode parameter of the IP Profile table to **Preference** or **Restriction and Preference**.

Outgoing SDP offer: The coders are arranged in the SDP offer according to the above if only allowed coders are used.

Notes:

If Allowed Coder Groups are configured, unknown coders are blocked by the device.

Allowed Coder Groups are applicable only to audio media.

For configuring Allowed Coder Groups, use the AllowedCodersGroup parameter.

The Allowed Coders process is as follows:

a. The device receives an incoming SIP message with SDP (offer) and checks the offered coders.

The source (first) leg may have Allowed Coders (i.e. list of coders that can be used - enforced).

The device checks for common coders between the SDP offered coders and the Allowed Coders Group list.

For example, assume the following:

The SDP coder offer includes the following coders: G.729, G.711, and G.723.

The source (first) leg includes the following Allowed Coders: G.711 and G.729.

The device selects the common coders, i.e., G.711 and G.729 (with changed preferred coder priority - highest for G.711). In other words, it removes the coders that are not in the Allowed Coders list and the order of priority is first according to the Allowed Coders list.

Now assume that the destination (second) leg also includes Allowed Coders. Therefore, the device performs the Allowed Coders procedure (common coders) between the updated coder list and the destination leg's Allowed Coders list. Adding to the example, assume the following:

For the first leg, the device selects the common coders G.711 and G.729 (explained in the example above).

Assume that the second leg includes the following Allowed Coders: G.723, G.726, and G.729.

As a result, the device selects the common coders, i.e., G.729 and G.726.

If the Allowed Coders policy on SDP returns an empty coders list, the device (source leg) rejects the call (SIP 488 or ACK and BYE).

Below is an example, assuming that Allowed Coders list (ordered) includes G711A-law (PCMA), G729, and G711U-law (PCMU).

1. SDP offer - original offer:

```
m=audio 6050 RTP/AVP 0 8 4 96
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:4 G723/8000
a=fmtp:4 annexa=no
a=rtpmap:96 telephone-event/8000
a=fmtp:96 0-15
a=ptime:20
a=sendrecv
```

SDP offer - after manipulation:

```
m=audio 6010 RTP/AVP 8 0 96
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:96 telephone-event/8000
a=fmtp:96 0-15
a=ptime:20
a=sendrecv
a=rtpmap:18 G729/8000
a=fmtp:18 annexb=no
```

In the SDP, the "m=audio 6010 RTP/AVP 8 0 96 18" line shows that the coder priority has changed - G.711A-law ("8") and then G.711U-law ("0"). The G.723 coder ("4) in the original offer was removed as it was not defined in the Allowed Coders list (i.e., a restricted coder).

18.1.5.5 SRTP-RTP Transcoding

The device supports transcoding between SRTP and RTP. The device can also enforce SBC legs to use SRTP\RTP, using the IP Profile parameter SBCMediaSecurityBehaviour:

As is (default): no special handling for RTP\SRTP is done.

SRTP: SBC legs negotiate only SRTP media lines, and RTP media lines are removed from the incoming SDP offer\answer.

RTP: SBC legs negotiate only RTP media lines, and SRTP media lines are removed from the incoming offer\answer.

Both: each offer\answer is extended (if not already) to two media lines - one RTP and the other SRTP.

If two SBC legs (after offer\answer negotiation) use different security types (i.e., one RTP and the other SRTP), then the device performs RTP-SRTP transcoding.

To transcode between RTP and SRTP, the following prerequisites must be met:

At least one supported SDP "crypto" attribute and parameters

EnableMediaSecurity must be set to 1

If one of the above transcoding prerequisites is not met:

Any value other than “As is” is discarded.

18.1.5.6 Multiple RTP Media Streams per Call Session

The device's SBC application supports multiple RTP media streams per SBC call session. Up to five different media types can be included in a session:

Audio (m=audio)

Video (m=video)

Text (m=text)

Fax (m=image)

18.1.6 Fax Negotiation and Transcoding

The device can allow fax transmissions to traverse transparently (i.e., without transcoding) or it can handle the fax as follows:

Allow interoperability between different fax machines, supporting fax transcoding if required.

Restrict usage of specific fax coders to save bandwidth, enhance performance, or comply with supported coders. These coders include G.711 (A-Law or Mu-Law), VBD (G.711 A-Law or G.711 Mu-Law), and T38.

Fax configuration is done in the IP Profile and Coder Group Settings tables. The IP Profile table determines the supported fax coders and the negotiation method used between the incoming and outgoing fax legs, using the following fax-related parameters:

SBCFaxBehavior: defines the offer negotiation method - pass fax transparently, negotiate fax according to fax settings in IP Profile, or enforce remote UA to first establish a voice channel before fax negotiation.

SBCFaxCodersGroupID: defines the supported fax coders (from the Coders Group Settings table).

SBCFaxOfferMode: determines the fax coders sent in the outgoing SDP offer.

SBCFaxAnswerMode: determines the fax coders sent in the outgoing SDP answer.

Notes:

Currently, FAX transcoding is not supported.

The voice-related coder configuration (Allowed coders) is independent of the fax-related coder configuration, with the exception of the G.711 coder. If the G.711 coder is restricted by the Allowed Coders Group table, it is not used for fax processing even if it is listed in the Coders Group Settings table for faxes. However, support for G.711 coders for voice is not dependent upon which fax coders are listed in the Coders Group Settings table.

18.1.7 SIP Dialog Admission Control

The device allows you to limit the number of concurrent calls (SIP dialogs). These call limits can be applied per SRD and/or IP Group, and per user (identified by its registered contact).This feature can be useful for implementing Service Level Agreements (SLA) policies.

The SIP dialog limits can be defined per SIP request type and direction (inbound or outbound). These relate to requests that initiate SIP dialogs and not the subsequent requests that can be of different type and direction. The SIP dialog-initiating request types can include SIP INVITEs, REGISTER, and/or SUBSCRIBE, or it can be configured to include all dialogs. Requests that supersede the defined limit are rejected with a SIP 486 "Busy Here" response.

SIP-dialog rate control can also be configured using the “token bucket” mechanism. The token bucket is a control mechanism that dictates the rate of SIP-dialog setups based on the presence of tokens in the bucket – a logical container that holds aggregate SIP dialogs to be accepted or transmitted. Tokens in the bucket are removed ("cashed in") for the ability to setup a dialog. Therefore, a flow can set up dialogs up to its peak burst rate if there are adequate tokens in the bucket and if the burst threshold is configured appropriately:

Every SIP dialog setup request must attempt to take a token from the bucket.

If there are no tokens, the request is dropped.

New tokens are added to the bucket at a user-defined rate (token rate).

If the bucket contains the maximum number of tokens, tokens to be added at that moment are dropped.

A token bucket is configured using the following new parameters:

Rate = Rate at which tokens are added to the bucket (i.e., token rate). One token is added to the bucket every 1000/Rate milliseconds. The rate of dialog setups per second or unlimited if set to 0 (default).

Max Burst = Maximum tokens that can fill the bucket. At any given time, the bucket cannot contain more than this amount of tokens. The maximum burst size for the dialog setup rate, unlimited if set to 0 (default).

Dropped requests are replied with the 486 “Busy Here” SIP response. Dropped requests are not counted in the bucket.

The SIP dialog limits are defined in the Admission Control table (SBCAdmissionControl).

18.1.8 Limiting SBC Call Duration

You can define a maximum allowed duration (in minutes) for SBC calls. If an established call reaches this user-defined limit, the device terminates the call. This feature ensures calls are properly terminated, allowing available resources for new calls. This feature is configured using the MaxCallDuration parameter.

18.1.9 SIP Authentication Server for SBC Users

The device can function as an authentication server for SIP SBC message requests, based on HTTP authentication DIGEST with MD5. Alternatively, such requests can be authenticated by an external, third-party server.

When functioning as an authentication server (set by the IP Group table parameter, AuthenticationMode), the device authenticates users belonging to a USER-type IP Group. When the device receives an INVITE or REGISTER request from a client (e.g., SIP phone) for SIP message authorization, the device processes the authorization as follows:

1. The device verifies the type of incoming SIP method (e.g., INVITE) that must be challenged for authorization. This is configured using the IP Group table parameter, MethodList.

If the message is received without an Authorization header, the device "challenges" the client by sending a 401 or 407 SIP response. The client then resends the request with an Authorization header (containing the user name and password).

The device validates the SIP message according to the settings of the parameters, AuthNonceDuration, AuthChallengeMethod and AuthQOP.

If validation fails, the message is rejected and the device sends a 403 "Forbidden" response.

If validation succeeds, the device verifies identification of the SBC user. This is done by checking that the user name and password received from the user is the same username and password that appears in the device's database. The SBC users in the database are obtained from the User Information file. If the SIP SBC user is not successfully authenticated after three attempts, the device sends a 403 "Forbidden" response.

If the user is successfully identified, the SIP message request is processed.

18.1.10 Handling SIP 3xx Redirect Responses

By default, the device's handling of SIP 3xx responses is to send the Contact header unchanged. However, some network setups require that the new INVITE message sent as a result of the 3xx traverse the device. This is enabled by the parameter SBC3xxBehavior.

Reasons for enforcing resultant INVITEs to traverse the SBC may vary:

The user that receives the 3xx can’t route to the 3xx contact (i.e., the user is on the LAN and the new contact is on the WAN). In such a scenario, the device helps the user reach the WAN contact and overcome NAT problems.

Enforce certain SBC policies (e.g., call admission control, header manipulation, and

transcoding) on the resultant INVITE.

The device enforces this by modifying each Contact in the 3xx response as follows:

Changes the host part to the device's IP address – this change causes the remote user agent to send the INVITE to the device.

Adds a special prefix ("T~&R_") to the Contact user part – to identify the new INVITE as a 3xx resultant INVITE.

The SBC handling for the 3xx resultant INVITE is as follows:

1. The incoming INVITE is identified as a 3xx resultant INVITE according to the special prefix.

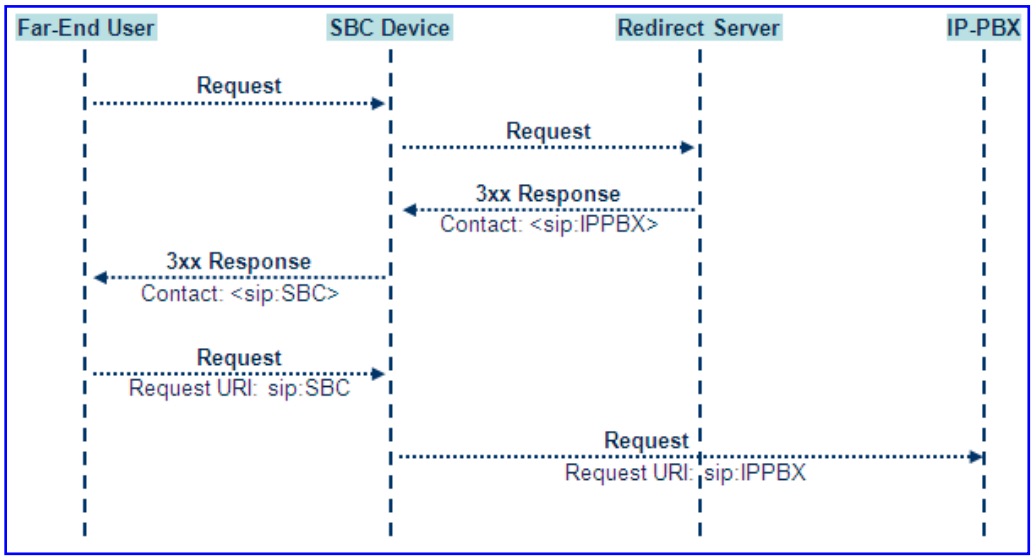
The device automatically replaces the SBC host part (in the Request-URI) with the host from the 3xx Contact.

The prefix ("T~&R_") remains in the user part for the classification, manipulation, and routing mechanisms.

The classification, manipulation, and routing processes are done exactly like any other INVITE handling. The special prefix can be used for specific routing rules for 3xx resultant INVITEs.

The prefix is removed before the resultant INVITE is sent to the destination.

Figure 18-9: SIP 3xx Response Handling



The process of this feature is described using an example:

1. The device receives the Redirect server's SIP 3xx response (e.g., Contact: <sip:User@IPPBX:5060;transport=tcp;param=a>;q=0.5).

The device replaces the Contact header value with the special prefix and database key value as user part, and with the device's URL as host part (e.g., Contact: <sip:Prefix_Key_User@SBC:5070;transport=udp>;q=0.5).

The device sends this manipulated SIP 3xx response to the Far-End User (FEU).

The FEU sends a new request with the Request-URI set to the value of the received 3xx response's Contact header (e.g., RequestURI: sip:Prefix_Key_User@SBC:5070;transport=udp).

Upon receipt of the new request from the FEU, the device replaces the Request-URI with the new destination address (e.g., RequestURI: sip:Prefix_User@IPPBX:5070;transport=tcp;param=a).

The device removes the user prefix from the Request-URI, and then sends this Request-URI to the new destination (e.g., RequestURI: sip:User@IPPBX:5070;transport=tcp;param=a).

18.1.11 Interworking SIP Diversion and History-Info Headers

This device can be configured to interwork between the SIP Diversion and History-Info headers. This is important, for example, to networks that support the Diversion header but not the History-Info header, or vice versa. Therefore, mapping between these headers is crucial for preserving the information in the SIP dialog regarding how and why (e.g., call redirection) the call arrived at a certain SIP UA.

This feature is configured in the IP Profile table (IPProfile parameter) using the following new parameters:

SBCDiversionMode - defines the device's handling of the Diversion header

SBCHistoryInfoMode - defines the device's handling of the History-Info header

The handling of the SIP Diversion and History-Info headers is described in the table below:

Table 18-1: Handling of SIP Diversion and History-Info Headers

Parameter Value	SIP Header Present in Received SIP Message		
	Diversion	**History-Info**	**Diversion and History-Info**
HistoryInfoMode = Add **DiversionMode = Remove**	Diversion converted to History-Info. Diversion removed.	Not present	Diversion removed.
HistoryInfoMode = Remove **DiversionMode = Add**	Not present.	History-Info converted to Diversion. History-Info removed.	History-Info added to Diversion. History-Info removed.
HistoryInfoMode = Disable **DiversionMode = Add**	Diversion converted to History-Info.	Not present.	Diversion added to History-Info.
HistoryInfoMode = Disable **DiversionMode = Add**	Not present.	History-Info converted to Diversion.	History-Info added to Diversion.
HistoryInfoMode = Add **DiversionMode = Add**	Diversion converted to History-Info.	History-Info converted to Diversion.	Headers are synced and sent.
HistoryInfoMode = Remove **DiversionMode = Remove**	Diversion removed.	History-Info removed.	Both removed.

18.1.12 Call Survivability

This section describes various call survivability features supported by the SBC device.

18.1.12.1 Auto-Provisioning of Subscriber-Specific Information for BroadWorks Server for Survivability

This feature enables SBC user registration for interoperability with BroadSoft BroadWorks server to provide call survivability in case of connectivity failure with the BroadWorks server, for example, due to a WAN failure. This feature enables local users to dial a local extension (or any other configured alias) that identifies another local user, in survivability mode. This feature is enabled using the SBCExtensionsProvisioningMode parameter.

In normal operation, when subscribers (such as IP phones) register to the BroadWorks server through the device, the device includes the SIP Allow-Events header in the sent REGISTER message. In response, the BroadWorks server sends the device a SIP 200 OK containing an XML body with subscriber information such as extension number, phone number, and URIs (aliases). The device forwards the 200 OK to the subscriber (without the XML body).

Figure 18-10: Interoperability with BroadWorks Registration Process

The device saves the users in its registration database with their phone numbers and extensions, enabling future routing to these destinations during survivability mode. When in survivability mode, the device routes the call to the Contact associated with the dialed phone number or extension number in the registration database.

Below is an example of an XML body received from the BroadWorks server:

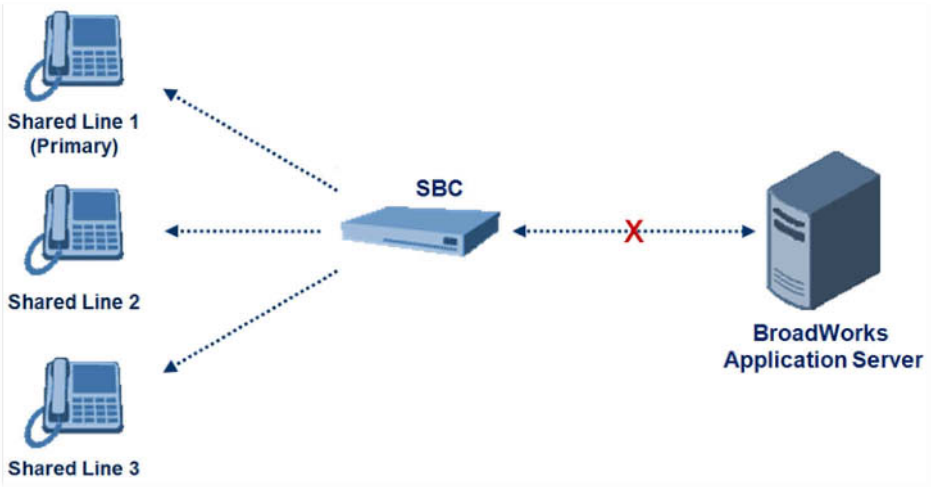
```
<?xml version="1.0" encoding="utf-8"?>
<BroadsoftDocument version="1.0" content="subscriberData">
<phoneNumbers>
<phoneNumber>2403645317</phoneNumber>
<phoneNumber>4482541321</phoneNumber>
</phoneNumbers>
<aliases>
<alias>sip:bob@broadsoft.com</alias>
<alias>sip:rhughes@broadsoft.com</alias>
</aliases>
<extensions>
<extension>5317</extension>
<extension>1321</extension>
</extensions>
</BroadSoftDocument>
```

18.1.12.2 BroadSoft's Shared Phone Line Call Appearance for SBC Survivability

The device can provide redundancy for BroadSoft's Shared Call Appearance feature. When the BroadSoft application server switch (AS) fails or does not respond, or when the network connection between the device and the BroadSoft AS is down, the device manages the Shared Call Appearance feature for the SIP clients.

This feature is supported by configuring a primary extension and associating it with secondary extensions (i.e., *shared lines*) so that incoming calls to the primary extension also ring at the secondary extensions. The call is established with the first extension to answer the call, and consequently, the ringing at the other extensions stop. For example, assume primary extension number 600 is shared with secondary extensions 601 and 602. In the case of an incoming call to 600, all three phone extensions ring simultaneously (using the device's call forking feature as described in 'SIP Forking Initiated by SIP Proxy Server' on page 171). Note that incoming calls specific to extensions 601 or 602 ring only at these specific extensions.

Figure 18-11: Call Survivability for BroadSoft's Shared Line Appearance



To configure this capability, you need to configure a shared-line, inbound manipulation rule for registration requests to change the destination number of the secondary extension numbers (e.g. 601 and 602) to the primary extension (e.g., 600). In addition, call forking must also be enabled. The procedure below describes the main configuration required.

Notes:

The device enables outgoing calls from all equipment that share the same line simultaneously (usually only one simultaneous call is allowed per a specific shared line).

The LED indicator of a shared line may display the wrong current state.

To configure the Shared Line feature:

1. Create a SERVER-type IP Group for the BroadWorks server.

Create a USER-type IP Group for the IP phone users and set the EnableSBCClientForking to enable so that the device forks incoming calls to all contacts under the same AOR that are registered in the device's registration database.

Create a call routing rule in the IP2IP Routing table for routing calls between the above configured IP Groups.

Create a manipulation rule in the IP to IP Inbound Manipulation table for the secondary extensions (e.g., 601 and 602) so that they also register in the device's database under the primary extension (e.g., 600) contact:

Set the 'Manipulation Purpose' field to **Shared Line**.

Set the 'Source IP Group' field to the IP Group ID that you created for the users (e.g., 2).

Set the 'Source Username Prefix' field to represent the secondary extensions (e.g., 601 and 602).

Set the 'Manipulated URI' field to **Source** to manipulate the source URI.

Set the 'Remove From Right' field to "1" to remove the last digit of the extensions (e.g., 601 is changed to 60).

Set the 'Suffix to Add' field to "0" to add 0 to the end of the manipulated number (e.g., 60 is changed to 600).

Figure 18-12: Manipulation Rule for Shared Line

Index	Is Additional Manipulation	Manipulation Purpose	Source Ip Group	Source Username Prefix	Source Host	Destination Username Prefix
1	0	Shared Line	2	60(1-2)	*	*

Destination Host	Request Type	Manipulated URI	Remove From Left	Remove From Right	Leave From Right	Prefix to Add	Suffix to Add
*	All	Source	0	1	255		0

18.1.12.3 Call Survivability for Call Centers

The device supports call survivability for call centers. When a communication failure (e.g., in the network) occurs with the remote voice application server responsible for handling the call center application (such as IVR), the device routes the incoming calls received from the customer (i.e., from the TDM gateway) to the call center agents.

In normal operation, the device registers the agents in its users registration database. Calls received from the TDM gateway are forwarded by the device to the application server, which processes the calls and sends them to specific call center agents, through the device. Upon a failure with the application server, the device routes the calls from the TDM Gateway to the agents. The device routes the call to the first available user it founds. If the call is not answered by the user, the device routes it to the next available user. The SBC can handle a sequence of up to five users, after which the session is timed out and the call is dropped.

Figure 18-13: Normal Operation in Call Center Application

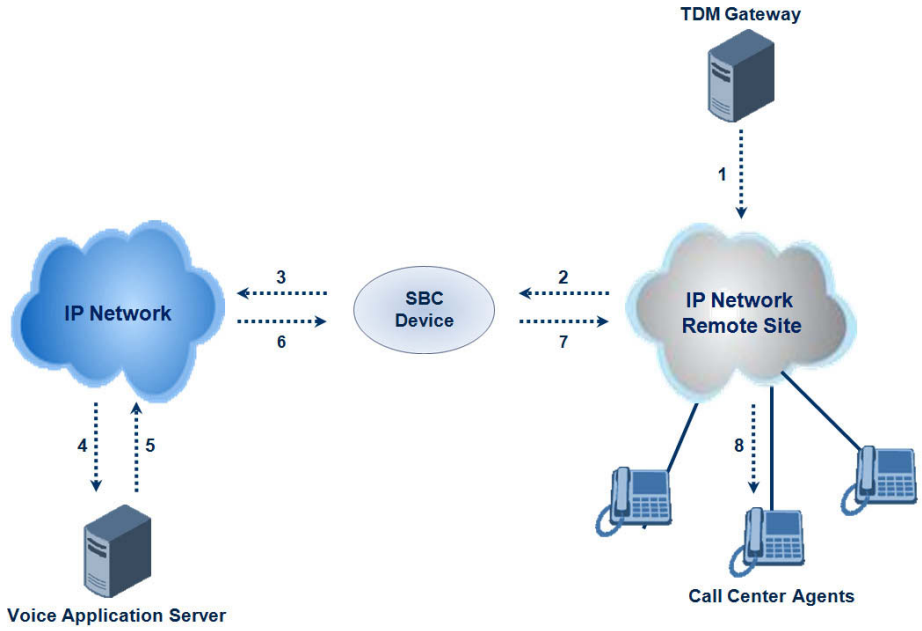
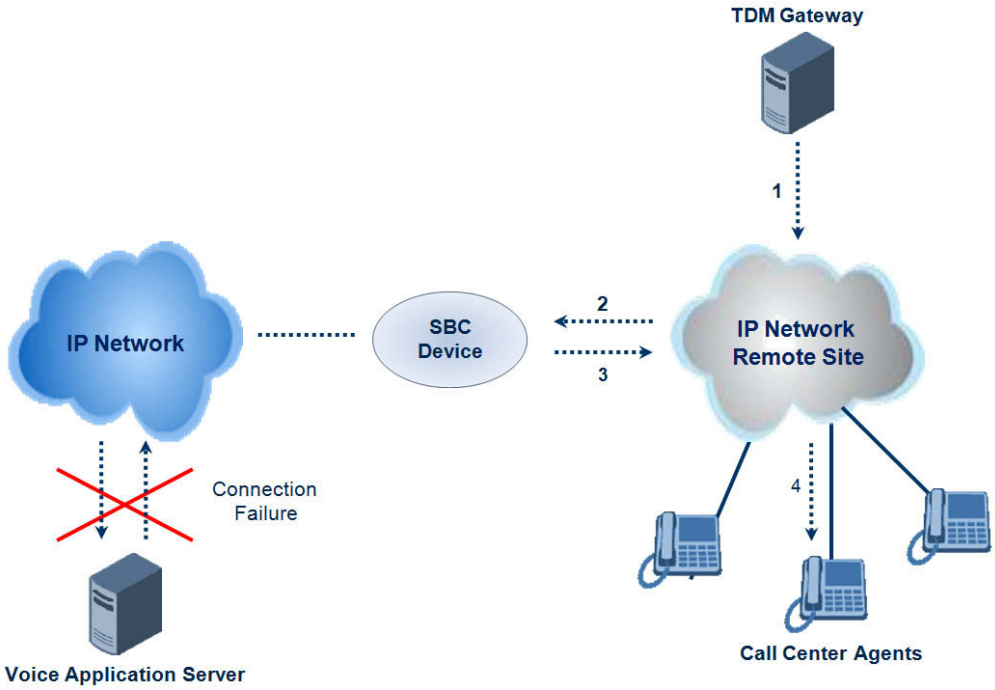


Figure 18-14: Call Survivability for Call Center



To configure call survivability for a call center application:

1. Configure IP Groups in the IP Group table (see 'Configuring IP Groups' on page 114) for the following entities:

   TDM Gateway (SERVER-type IP Group). This entity forwards the customer calls, through the device, to the Application server.

   Application server (SERVER-type IP Group). This entity processes the call and sends the call, through the device, to the specific call center agent located on a different network (remote).

   Call center agents (USER-type IP Group). You can configure multiple IP Groups to represent different groups of call center agents, for example, agents and managers.

In the Classification table (see 'Configuring Classification Table' on page 178), configure rules to classify incoming calls received from the entities listed in Step 1 to IP Groups.

In the SBC IP2IP Routing table (see 'Configuring SBC IP-to-IP Routing' on page 183), configure the following IP-to-IP routing rules:

For normal operation:

Routing from TDM Gateway to Application server.

Routing from Application server to call center agents.

For call survivability mode: Routing from TDM Gateway to call center agents. This configuration is unique due to the following settings:

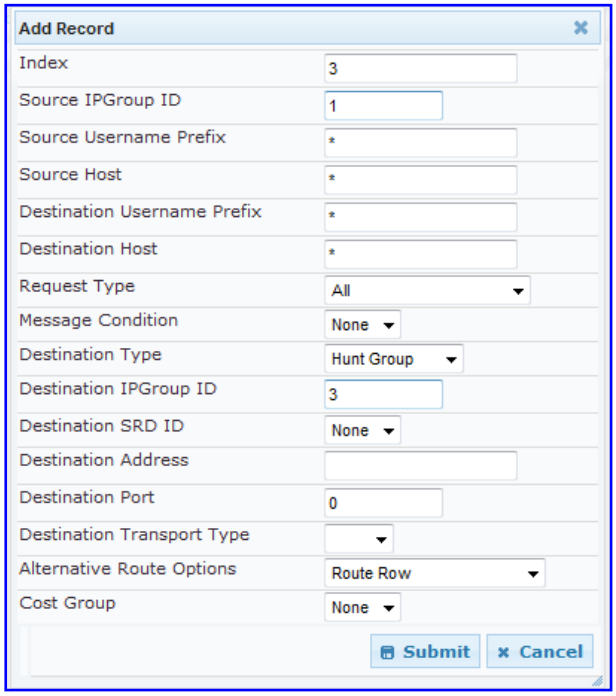
The 'Source IP Group ID' field is set to the IP Group of the TDM Gateway.

The 'Destination Type' field is set to **Hunt Group**, which is specifically used for call center survivability.

The 'Destination IP Group ID' field is set to the IP Group of the call center agents.

The figure below displays a routing rule example, assuming IP Group "1" represents the TDM Gateway and IP Group "3" represents the call center agents:

Figure 18-15: Routing Rule for Call Survivability of Call Center



18.1.12.4 Survivability Mode Display on Aastra IP Phones

If the SBC device is deployed in an Enterprise network with Aastra IP phones and connectivity with the WAN fails, the device provides call survivability by enabling communication between IP phone users within the LAN enterprise. In such a scenario, the device can be configured to

notify the IP phones that it is currently operating in Survivability mode. When this occurs, the Aastra IP phones display the message, "StandAlone Mode" on their LCD screens. This feature is enabled by setting the SBCEnableAASTRASurvivabilityNotice parameter to 1.

When this feature is enabled and the SBC device is in Survivability mode, it responds to SIP REGISTER messages from the IP phones with a SIP 200 OK containing the following XML body:

```
Content-Type: application/xml
<?xml version="1.0" encoding="utf-8"?>
<LMIDocument version="1.0">
<LocalModeStatus>
<LocalModeActive>true</LocalModeActive>
<LocalModeDisplay>StandAlone Mode</LocalModeDisplay>
</LocalModeStatus>
</LMIDocument>
```

18.1.13 Call Forking

18.1.13.1 Initiating SIP Call Forking

The SBC device enables call forking, whereby an incoming call is forked to multiple SBC users (destinations). In such a scenario, upon an incoming call, all the extensions of a user ring simultaneously and the first extension to pick up the call receives the call and all other extensions stop ringing. Call forking is supported by the device's ability to register in its database multiple SIP client user phone contacts (mobile and fixed-line extensions) to the same Address of Record (AOR).

This feature can be implemented in the following example scenarios:

An enterprise Help Desk, where incoming customer calls are simultaneously sent to multiple customer service agent extensions.

An employee's phone devices, where the incoming call is simultaneously sent to multiple devices (e.g., to the employee's office phone and mobile SIP phone).

An enterprise reception desk, where an incoming call is simultaneously sent to multiple receptionists.

The Call Forking feature is configured by creating a USER-type IP Group with the EnableSBCClientForking parameter set to enabled (in the IP Group table).

18.1.13.2 SIP Forking Initiated by SIP Proxy Server

The device can handle SIP forking responses received from a proxy server in response to an INVITE sent by the device from a UA. In other words, received responses with a different SIP To header 'tag' parameter for the request forwarded by the device. This occurs in scenarios, for example, where a proxy server forks the INVITE request to several UAs, and hence, the SBC device may receive several replies for a single request.

Forked SIP responses may result in a single SDP offer with two or more SDP answers during call setup. The SBC handles this scenario by "hiding" the forked responses from the INVITE-initiating UA. This is achieved by marking the UA that responded first to the INVITE as the active UA, and only requests/responses from that UA are subsequently forwarded. All other requests/responses from other UAs are handled by the SBC (SDP offers from these users are answered with an 'inactive' media).

If the active UA is the first one to send the final response (e.g., 200 OK), the call is established and all other final responses are acknowledged and a BYE is sent if needed. If another UA sends the first final response, then it is possible that the SDP answer that was forwarded to the INVITE-initiating UA is not relevant, and media synchronization is needed between the two UAs. Media synchronization is done by sending a re-INVITE request immediately after the call is established. The re-INVITE is sent without an offer to the INVITE-initiating UA. This causes the UA to send an offer which is forwarded to the UA that confirmed the call. The media synchronization process is enabled by the EnableSBCMediaSync parameter.

18.1.14 Alternative Routing on Detection of Failed SIP Response

The device can detect failure of a sent SIP response (e.g., TCP timeout, and UDP ICMP). In such a scenario, the device re-sends the response to an alternative destination. This support is in addition to alternative routing if the device detects failed SIP requests.

For example, assume the device sends a SIP 200 OK in response to a received INVITE request. If the device does not receive a SIP ACK in response to this, it sends a new 200 OK to the next alternative destination (e.g., to the next given IP address resolved from a DNS from the Contact or Record-Route header in the request related to the response).

18.2 SBC Configuration

This section describes the configuration of the SBC application.

Note: For the SBC application, the following requirements must be met:

The SBC application is enabled (see 'Enabling Applications' on page 109).

The 'SBC' Software Upgrade Key is installed on the device (see 'Loading Software Upgrade Key' on page 250).

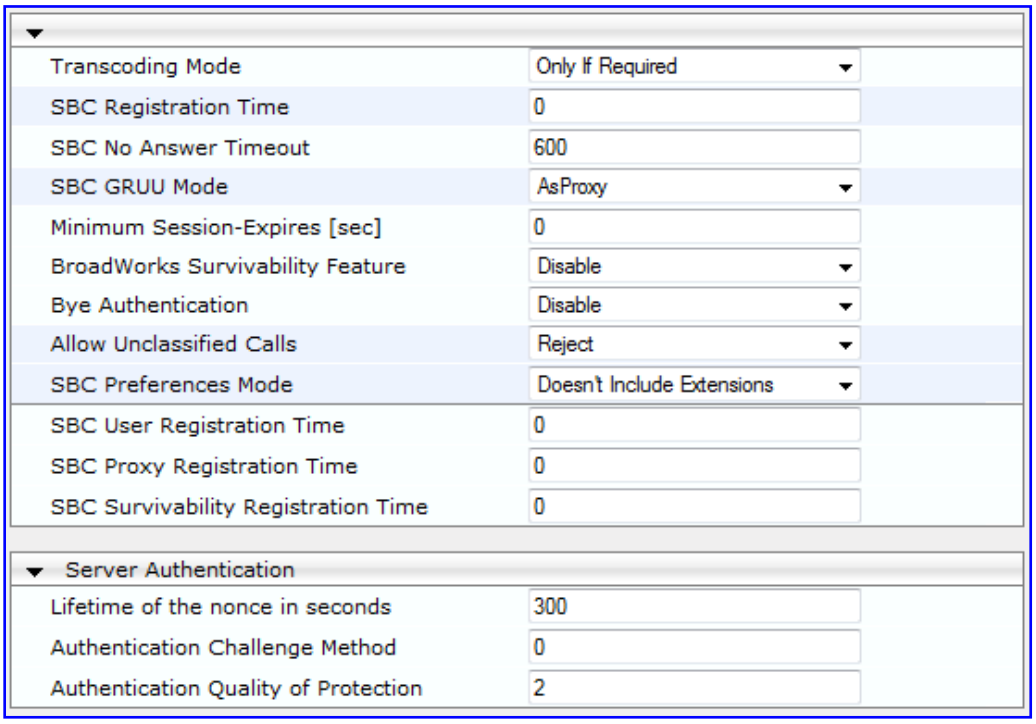
18.2.1 Configuring General Settings

The General Settings page allows you to configure general SBC parameters. For a description of these parameters, see 'SBC Parameters' on page 339.

To configure general parameters:

1. Open the General Settings page (**Configuration** tab >**VoIP** menu >**SBC** submenu >**General Settings**).

Figure 18-16: General Settings Page



Configure the parameters as required.

Click Subm**it to** apply your changes.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

18.2.2 Configuring Admission Control

The Admission Control page allows you to define up to 100 rules for limiting the number of concurrent calls (SIP dialogs). These call limits can be applied per SRD, IP Group, SIP request type (e.g., INVITEs), SIP dialog direction (e.g., inbound), and/or per user (identified by its registered contact). This feature can be useful for implementing Service Level Agreements (SLA) policies.

The SIP dialog limits can be defined per SIP request type and direction. These relate to requests that initiate SIP dialogs and not the subsequent requests that can be of different type and direction. The SIP dialog-initiating request types can include SIP INVITEs, REGISTER, and/or SUBSCRIBE, or it can be configured to include the total number of all dialogs.

This feature also provides support for SIP-dialog rate control, using the “token bucket” mechanism. The token bucket is a control mechanism that dictates the rate of SIP-dialog setups based on the presence of tokens in the bucket – a logical container that holds aggregate SIP dialogs to be accepted or transmitted. Tokens in the bucket are removed ("cashed in") for the ability to setup a dialog. Therefore, a flow can setup dialogs up to its peak burst rate if there are adequate tokens in the bucket and if the burst threshold is configured appropriately.

Requests that reach the user-defined call limit (maximum concurrent calls and/or call rate) are sent to an alternative route, if configured in the IP2IP Routing table for the SRD or IP Group. If no alternative routing rule is located, the device rejects the SIP request with a SIP 486 "Busy Here" response.

Notes:

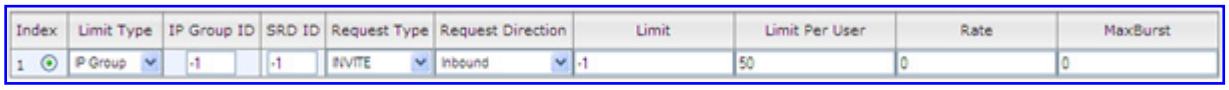
The enforcement of a configured limitation for the incoming leg is performed immediately after the Classification process. If the call/request is rejected at this stage, no routing is performed. The enforcement for the outgoing leg is performed within each alternative route iteration. This is accessed from two places: one during initial classification/routing, and another during alternative routing process.

For configuring Admission Control using the *ini* file, refer to the parameter SBCAdmissionControl.

To configure Admission Control rules:

1. Open the Admission Control page (**Configuration** tab >**VoIP** menu >**SBC** submenu >**Admission Control**).

Figure 18-17: Admission Control Page



Add an entry and then configure it according to the table below.

Click the **Apply** button to save your changes.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

Table 18-2: Admission Control Parameters

Parameter	Description
Limit Type	Limitation rule defined per IP group or SRD. **[0]** IP Group (default) **[1]** SRD
IP Group ID	IP Group to which you want to apply the SIP dialog limit. To apply the rule to all IP Groups, set this parameter to -1 (default). **Note:** This parameter is applicable only if Limit Type is set to IP Group.
SRD ID	SRD to which you want to apply the SIP dialog limit. To apply the rule to all SRD's, set this parameter to -1 (default). **Note:** This parameter is applicable only if Limit Type is set to SRD.
Request Type	SIP dialog-initiating request type that initiates the SIP dialog to which you want to apply the SIP dialog limit (not the subsequent requests that can be of different type and direction). The SIP dialog-initiating request types can include: **[0]** All = include the total number of all dialogs (default) **[1]** INVITE **[2]** SUBSCRIBE **[3]** Other
Request Direction	The direction of the SIP request to which the limitation is applied. **[0]** Both = Applied to inbound and outbound SIP dialogs (default) **[1]** Inbound = Applies only to inbound SIP dialogs **[2]** Outbound = Applies only to outbound SIP dialogs
Limit	Maximum number of concurrent SIP dialogs per IP Group or SRD. You can also use the following special values: **[0]** 0 = Disallow/block all these dialogs **[-1]** -1 = No limit (default)
Limit Per User	Maximum number of concurrent SIP dialogs per user belonging to the configured IP Group or SRD. You can also use the following special values: **[0]** 0 = Disallow/block all these dialogs **[-1]** -1 = No limit (default)
Rate	Rate at which tokens are added to the bucket per second (i.e., token rate) or unlimited if set to 0 (default). One token is added to the bucket every 1000 divided by the value of this parameter (in milliseconds). **Note:** The token bucket feature is per IP Group, SRD, SIP request type, and SIP request direction.
MaxBurst	The maximum number of tokens (SIP dialogs) that the bucket can hold, where 0 is unlimited (default). The device only accepts a SIP dialog if a token exists in the bucket. Once the SIP dialog is accepted, a token is removed from the bucket. If a SIP dialog is received by the device and the token bucket is empty, then the device rejects the SIP dialog. Alternatively, if the bucket is full, for example, 100 tokens, and 101 SIP dialogs arrive (before another token is added to the bucket, i.e., faster than that defined in the Rate field), then the device accepts the first 100 SIP dialogs and rejects the last one. Dropped requests are replied with the 486 “Busy Here” SIP response. Dropped requests are not counted in the bucket. **Note:** The token bucket feature is per IP Group, SRD, SIP request type, and SIP request direction.

18.2.3 Configuring Allowed Coder Groups

The Allowed Coders Group page allows you to define up to five Allowed Coder Groups, each with up to 10 coders. Allowed Coder Groups determine the coders that can be used for a specific SBC leg. Therefore, the device's SBC application can enforce the use of specific coders while preventing the use of other coders. Coders excluded from the Allowed Coders Group are removed from the SDP offer. Only common coders between SDP offered coders and coders configured in the Allowed Coder Groups are used.

The order of appearance of coders in the Allowed Coder Group determines the coder priority (preference), whereby the first coder is given the highest priority. For more information on working with Allowed and Extension coders, and coder preference, see 'Restricting Coders' on page 157.

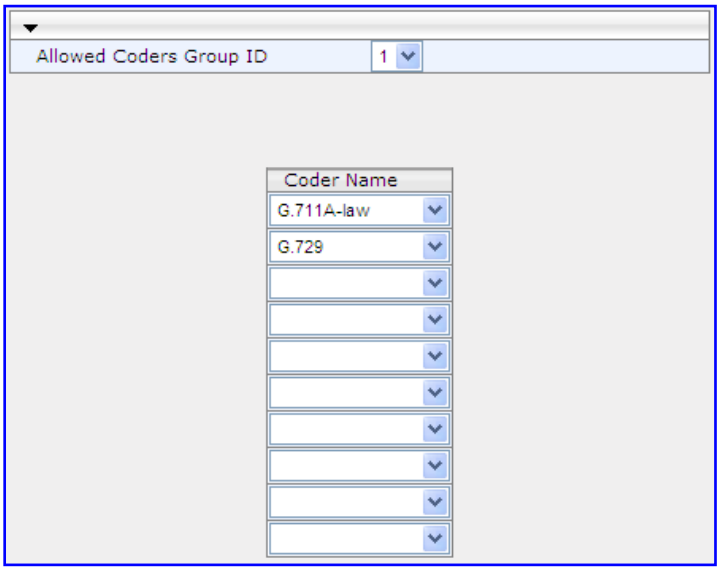
Notes:

- Each coder can appear only once per Allowed Coder Group.
- If Allowed Coder Groups are configured, coders not included are blocked by the device.
- Allowed Coder Groups are applicable only to audio media.
- Allowed Coder Groups can be assigned to IP Profiles (see 'Configuring IP Profiles' on page 137).
- You can also configure Allowed Coder Groups using the *ini* file parameter table AllowedCodersGroup.

To configure Allowed Coder Groups:

1. Open the Allowed Coders Group page (**Configuration** tab >**VoIP** menu >**SBC** submenu >**Allowed Coders Group**).

Figure 18-18: Allowed Coders Group Page



From the 'Allowed Coders Group ID' drop-down list, select an ID for the Allowed Coder Group.

In the Coder Name table, select coders for the Allowed Coder Group.

Click Submit to apply your changes.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

18.2.4 Configuring SIP Message Policy Rules

You can configure SIP message policies for blocking (blacklist) unwanted incoming SIP messages and allowing (whitelist) receipt of desired messages. This feature allows you to define legal and illegal characteristics of a SIP message. The message policy can apply globally (default) or per signaling domain (i.e., assigned to a SIP interface in the SIP Interface table - see 'Configuring SIP Interface Table' on page 112).

This feature is helpful against VoIP fuzzing (also known as robustness testing), which sends different types of packets to its "victims" for finding bugs and vulnerabilities. For example, the attacker might try sending a SIP message containing either an over-sized parameter or too many occurrences of a parameter.

SIP message security rules are configured in the new Message Policy table (MessagePolicy). Each policy can be defined with the following:

Maximum message length

Maximum SIP header length

Maximum message body length

Maximum number of headers

Maximum number of bodies

Option to send 400 "Bad Request" response if message request is rejected

Blacklist and whitelist for defined SIP methods (e.g., INVITE)

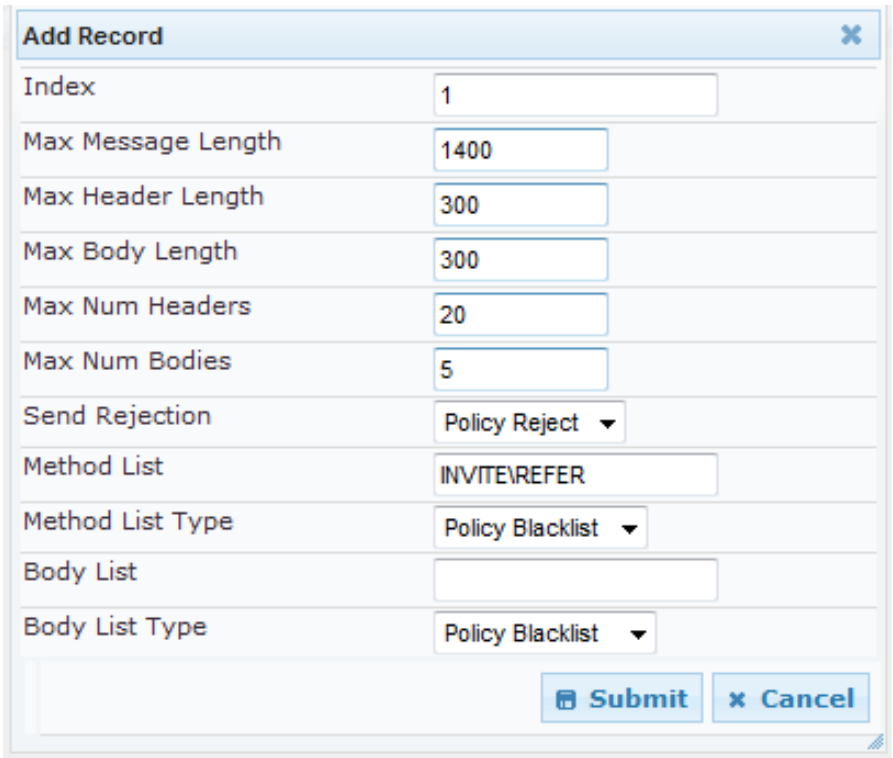
Blacklist and whitelist for defined SIP bodies

To configure SIP message policy rules:

1. Open the Message Policy Table page (**Configuration** tab >**VoIP** menu >**SBC** submenu >**Message Policy Table**).

Click the **Add** button; the Add Record dialog box appears:

Figure 18-19: Adding SIP Message Policy Rule



The policy defined in the previous figure limits messages to 32768 characters, headers to 256 characters, bodies to 512 characters, limits number of headers to 16, and only permits

two bodies. Invalid requests are rejected. Only INVITE and BYE requests are permitted and there are no restrictions on bodies.

Configure the SIP message policy rule as required. See the table below for a description of each parameter.

Click Submit to apply your changes.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

Table 18-3: SIP Message Policy Parameters

Parameter	Description
Index **[MessagePolicy_Index]**	Defines the table index entry.
Max Message Length **[MessagePolicy_MaxMessageLength]**	Defines the maximum SIP message length. The valid value is up to 32768 characters.
Max Header Length **[MessagePolicy_MaxHeaderLength]**	Defines the maximum SIP header length. The valid value is up to 256 characters.
Max Body Length **[MessagePolicy_MaxBodyLength]**	Defines the maximum SIP message body length. This is the value of the Content-Length header. The valid value is up to 512 characters.
Max Num Headers **[MessagePolicy_MaxNumHeaders]**	Defines the maximum number of headers. The valid value is any number up to 16.
Max Num Bodies **[MessagePolicy_MaxNumBodies]**	Defines the maximum number of bodies. The valid value is any number up to 2.
Send Rejection **[MessagePolicy_SendRejection]**	Determines whether the device sends a 400 "Bad Request" response if a message request is rejected. Policy Reject = If the message is a request, then the device sends a response to reject the request. Policy Drop = The device ignores the message without sending any response.
Method List **[MessagePolicy_MethodList]**	Defines the SIP methods (e.g., INVITE\BYE) to which the rule applies. The syntax for entering the methods is as follows: Methods must be separated by a backslash (\). The entered value is not case sensitive.
Method List Type **[MessagePolicy_MethodListType]**	Determines the policy for the SIP methods. Policy Blacklist = The specified methods (in the 'Method List' field) are rejected by the policy. Policy Whitelist = The specified methods (in the 'Method List' field) are allowed by the policy.
Body List **[MessagePolicy_BodyList]**	Defines the SIP body (i.e., value of the Content-Type header) to which the rule applies.
Body List Type **[MessagePolicy_BodyListType]**	Determines the policy for the defined SIP body. Policy Blacklist =The specified SIP body (in the 'Body List' field) is rejected by the policy. Policy Whitelist = The specified SIP body (in the 'Body List' field) is allowed by the policy.

18.2.5 Routing SBC

This section describes the configuration of the routing entities for the SBC application. These include the following:

Classification rules - see 'Configuring the Classification Table' on page 178

Condition rules - see 'Configuring Condition Rules' on page 182

IP-to-IP routing rules - see 'Configuring the IP-to-IP Routing' on page 183

Alternative routing reasons - see 'Configuring Alternative Routing Reasons' on page 189

18.2.5.1 Configuring Classification Rules

The Classification table allows you to configure classification rules. Classification rules are used to classify incoming SIP dialog-initiating requests (e.g., SIP INVITE messages) to source IP Groups from where the SIP dialog request originated. Classification rules also enhance security by allowing you to create a SIP access list of whitelists and blacklists. Incoming SIP dialog requests matching the user-defined classification characteristics can be denied (i.e., blacklist) or allowed (i.e., whitelist). The identified source IP Group is later used in manipulation and routing processes.

The Classification table is used to classify the incoming SIP dialog request only if classification based on the device's registration database and Proxy Set fails. The classification process is as follows:

1. Classification starts with the device's registration database, where it searches for a match by checking if the request arrived from a registered user in the database:

   Compares Contact header of the received SIP dialog to the Contact of the registered user

   Compares P-Asserted/From URL to the registered AOR

If the database search fails, the classification process proceeds with locating a Proxy Set associated with the IP address of the SIP dialog request, and then locating an IP Group associated with this Proxy Set (in the IP Group table). This classification stage is relevant only if enabled for the IP Group, using the ClassifyByProxySet parameter (see 'Configuring IP Groups' on page 114). If enabled, the device classifies requests arriving from the IP Group's Proxy Set as coming from this IP Group. The classification is done according to the Proxy IP list (in case of host names, then according to the dynamically resolved IP address list). Note that this classification is irrelevant in cases where multiple IP Groups use the same Proxy Set.

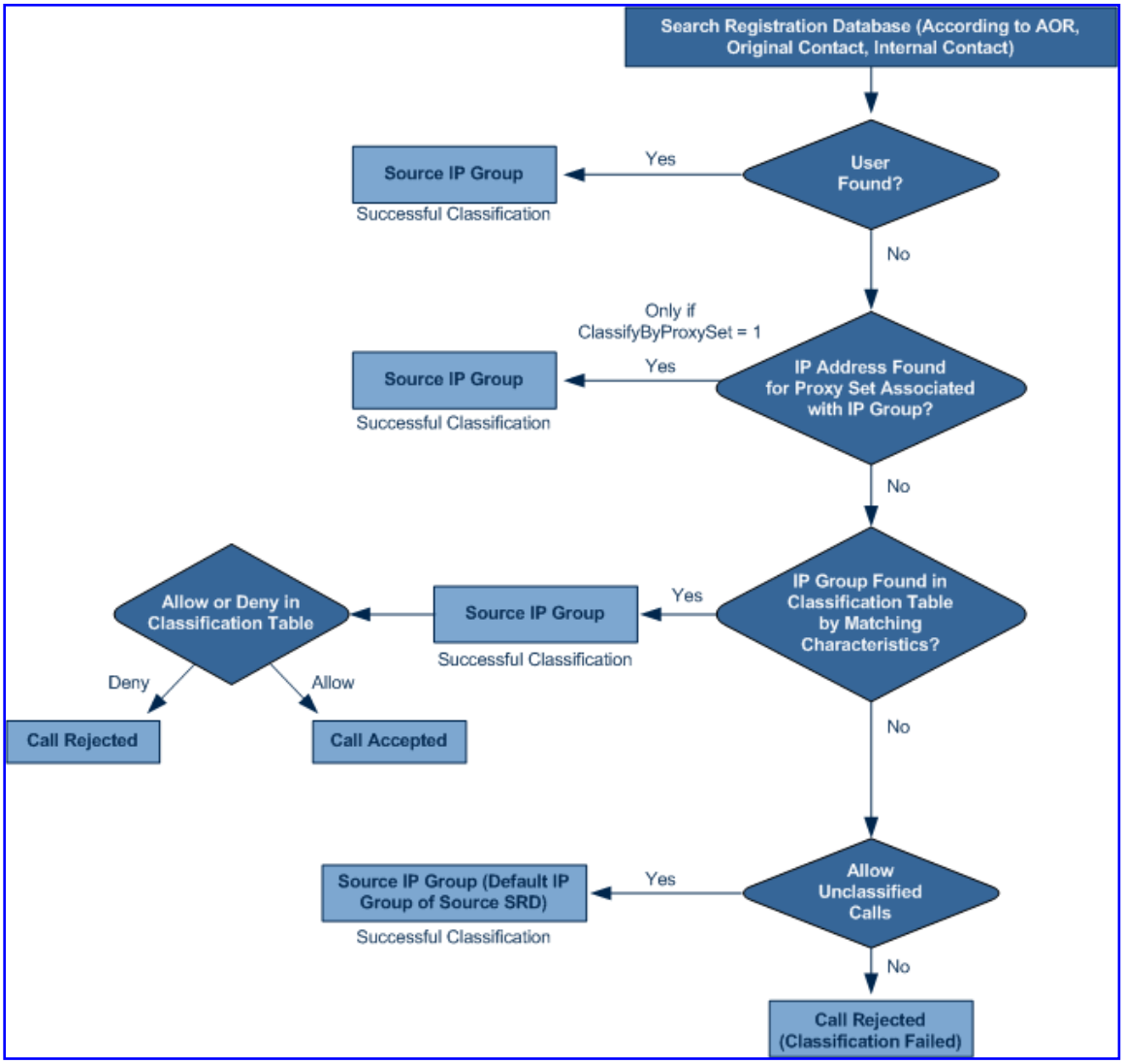
If classification based on Proxy Set fails, the device uses the Classification table to classify the SIP dialog to an IP Group. If it locates a classification rule whose characteristics (such as source IP address) match the incoming SIP dialog, then the SIP dialog is assigned to the associated IP Group. If the classification rule is defined as a whitelist, the SIP dialog is allowed and proceeds in the manipulation, routing and other processes. If the classification rule is defined as a blacklist, the SIP dialog is denied.

If the above classification process fails to determine the source IP Group to which the incoming packet belongs, the call is rejected or allowed, depending on the setting of the AllowUnclassifiedCalls parameter. If this parameter is enabled, the incoming SIP dialog is allowed and assigned to the default IP Group of the default SRD.

If the incoming SIP dialog cannot be classified according to the Classification table, the call is accepted or rejected based on the settings of the AllowUnclassifiedCalls parameter.

The flowchart below illustrates the classification process:

Figure 18-20: Classification Process (Identifying IP Group or Rejecting Call)



Notes:

Incoming REGISTER messages are saved in the device's registration database and sent to a destination only if they are associated with a source IP Group that is of USER type.

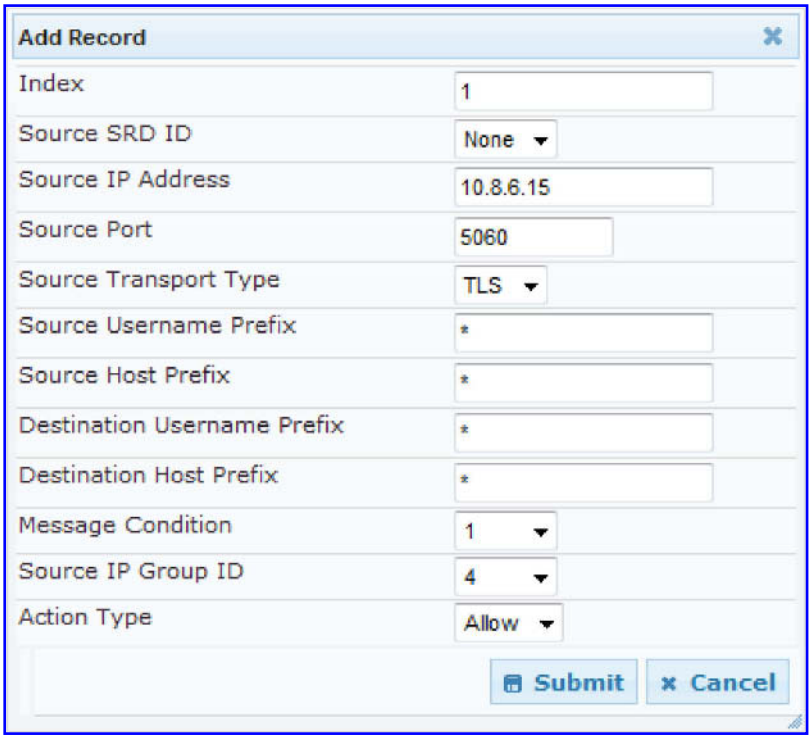
The Classification table can also be configured using the *ini* file table parameter Classification (see 'SBC Parameters' on page 339).

To configure classification rules:

1. Open the Classification Table page (**Configuration** tab >**VoIP** menu >**SBC** submenu >**Routing SBC** submenu >**Classification Table**).

Click the **Add** button; the following appears:

Figure 18-21: Classification Table Page



The figure above shows an example classification rule that identifies an incoming SIP dialog to IP Group ID #4, if its source IP address is 10.8.6.15, source port is 5060, SIP transport is TLS, and matches the Message Condition rule 1. The rule also allows (whitelist) this SIP dialog.

Configure the classification rule as required. For a description of the parameters, see the table below.

Click Subm**it to** apply your changes.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

Table 18-4: Classification Table Parameters

Parameter	Description
Index	Defines the index number of the table row entry.
Matching Characteristics	
Source SRD ID **[Classification_SrcSRDID]**	Selects the SRD ID of the incoming SIP dialog. The default is -1 (i.e., no SRD is assigned). **Notes:** The SRDs are configured in the SRD table (see 'Configuring SRD Table' on page 110). The SRDs are also associated with a port number as defined by the SIP Interface used by the SRD (see 'Configuring SIP Interface Table' on page 112).
Source IP Address	Defines the source IP address (in dotted-decimal notation) of the

Parameter	Description
[Classification_SrcAddress]	incoming SIP dialog. **Notes:** If this parameter is not configured or is configured as an '*' (asterisk), then any source IP address is accepted. The IP address can include the "x" wildcard to represent single digits. For example: 10.8.8.xx represents all the addresses between 10.8.8.10 to 10.8.8.99. The IP address can include the asterisk (*) wildcard to represent any number between 0 and 255. For example, 10.8.8.* represents all addresses between 10.8.8.0 and 10.8.8.255.
Source Port **[Classification_SrcPort]**	Defines the source port number of the incoming SIP dialog.
Source Transport Type **[Classification_SrcTransportType]**	Defines the source transport type (UDP, TCP, or TLS) of the incoming SIP dialog.
Source Username Prefix **[Classification_SrcUsernamePrefix]**	Defines the prefix of the source URI user part of the incoming SIP dialog. This is typically located in the SIP From URI. **Note:** The prefix can be a single digit or a range of digits. For available notations, see 'Dialing Plan Notation for Routing and Manipulation' on page 190.
Source Host Prefix **[Classification_SrcHost]**	Defines the prefix of the source URI host name in the From header of the incoming SIP dialog request. If this routing rule is not required, leave the field empty. The asterisk (*) symbol can be used to denote any source host prefix.
Destination Username Prefix **[Classification_DestUsernamePrefix]**	Defines the prefix of the destination Request-URI user part of the incoming SIP dialog. **Note:** The prefix can be a single digit or a range of digits. For available notations, see 'Dialing Plan Notation for Routing and Manipulation' on page 190.
Destination Host Prefix **[Classification_DestHost]**	Defines the prefix of the destination Request-URI host name of the incoming SIP dialog request. If this routing rule is not required, leave the field empty. The asterisk (*) symbol can be used to denote any destination host prefix.
Message Condition **[Classification_MessageCondition]**	Assigns a Condition rule which can also be used to classify the incoming SIP dialog. **Note:** Condition rules are configured in the Condition Table (see 'Configuring Condition Rules for Classification' on page 182).

Parameter	Description
Operation Rule	
Source IP Group ID **[Classification_SrcIPGroupID]**	Assigns an IP Group to the incoming SIP dialog request if this SIP dialog matches the matching rule. The default is -1 (i.e., no IP Group is assigned). **Notes:** The IP Group must be associated with the selected SRD. The IP Group is used for SBC routing and manipulations. To define IP Groups, see 'Configuring IP Groups' on page 114.
Action Type **[Classification_ActionType]**	Defines a whitelist or blacklist for incoming SIP dialog requests that match the characteristics of the classification rule. **[0]** Deny = Blocks incoming SIP dialogs that match the characteristics of the Classification rule (blacklist). **[1]** Allow = Allows incoming SIP dialogs that match the characteristics of the Classification rule (whitelist), and assigns it to the associated IP Group. (default)

18.2.5.2 Configuring Condition Rules

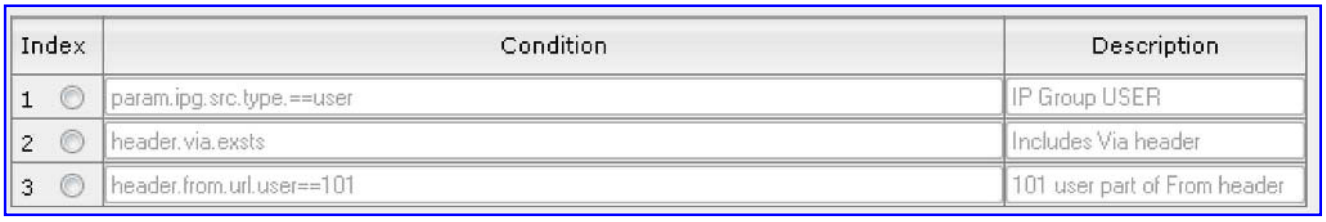
Condition rules allow you to enhance the process of classifying an incoming SIP dialog to an IP Group by using SIP message rules. Condition rules are later assigned to classification rules in the Classification table (see 'Configuring Classification Table' on page 178). When a classification rule is associated with a condition rule, the classification is used only if the classification rule and its associated condition rule are matched.

Condition rules are SIP message conditions configured using the same syntax (match-condition) as in the Message Manipulations table (for example, "header.to.host contains company"). You can also define complex rules using the "AND" or "OR" Boolean operands.

To configure condition rules:

1. Open the Condition Table page (**Configuration** tab >**VoIP** menu >**SBC** submenu >**Routing SBC** submenu >**Condition Table**).

Figure 18-22: Condition Table Page



The figure above shows the following configurations:

Index 1: Incoming SIP dialog that is classified as belonging to a USER-type IP Group.

Index 2: Incoming SIP dialog with a SIP Via header.

Index 3: Incoming SIP dialog with "101" as the user part in the SIP From header.

Click the Add button to add an entry, and then configure it according to the table below.

Click the **Apply** button to save your changes.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

Table 18-5: Condition Table Parameters

Parameter	Description
Condition **[ConditionTable**_Condition]	Defines the condition rule of the SIP message. The valid value is a string. For a description of the string syntax, see 'Syntax' on page 408. **Note:** Enclose user and host parts in single quotes.
Description **[ConditionTable**_Description]	Defines a brief description of the condition rule.

18.2.5.3 Configuring SBC IP-to-IP Routing

The IP2IP Routing Table page configures up to 120 SBC IP-to-IP routing rules. This table provides enhanced IP-to-IP call routing capabilities for routing received SIP dialog messages (e.g., INVITE) to a destination IP address. The SIP message is routed according to a routing rule whose configured input characteristics (e.g., Source IP Group) match the incoming SIP message. If the characteristics of an incoming call does not match the first rule, the call characteristics is then compared to those of the second rule, and so on until a matching rule is located. If no rule is matched, the call is rejected.

The IP-to-IP call destination can be one of the following:

Registered user Contact listed in the device's database (only for USER-type IP Groups).

Proxy Set associated with the destination IP Group (allows redundancy/load balancing).

Specific destination address (can be based on IP address, host name, port, transport type, and/or SRD). Routing to a host name can be resolved using NAPTR/SRV/A-Record.

Incoming Request-URI.

ENUM query.

For all destination types listed above except destination IP Group, the IP Group can optionally be itself, configured to provide the destination SRD and/or IP Profile. If neither destination SRD nor destination IP Group is defined, the destination SRD is the source SRD and the destination IP Group is its default IP Group.

In addition to the alternative routing/load balancing provided by the Proxy Set associated with the destination IP Group, the table allows the configuration of alternative routes whereby if a route fails, the next adjacent (below) rule in the table that is configured as 'Alt Route Ignore/Consider Inputs' are used. The alternative routes rules can be set to enforce the input matching criteria or to ignore any matching criteria.

Alternative routing occurs upon one of the following conditions:

A request sent by the device is responded with one of the following:

- SIP response code (i.e., 4xx, 5xx, and 6xx SIP responses) configured in the SBC Alternative Routing Reasons table (see 'Configuring Alternative Routing Reasons' on page 189).
- SIP 408 Timeout or no response (after timeout).

The DNS resolution includes IP addresses that the device has yet to try (for the current call).

Messages are re-routed with the same SIP Call-ID and CSeq header fields (increased by 1).

Notes:

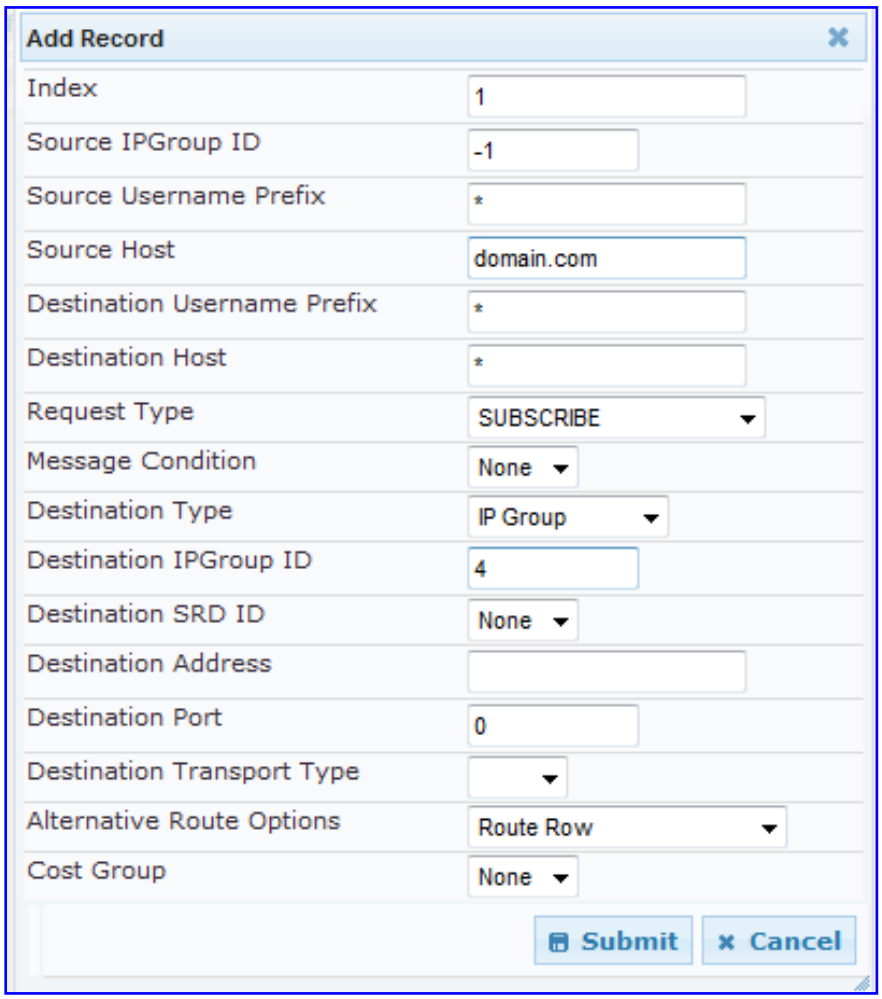
- For a specific IP-to-IP routing rule to be effective, the incoming SIP dialog message must match the characteristics configured for that rule.
- The IP2IP Routing table can also be configured using the *ini* file table parameter IP2IPRouting (see 'SBC Parameters' on page 339).

To configure SBC IP-to-IP routing rules:

1. Open the IP2IP Routing Table page (**Configuration** tab >**VoIP** menu >**SBC** submenu >**Routing SBC** submenu >**IP to IP Routing Table**).

Click the **Add** button; the Add Record dialog box appears:

Figure 18-23: SBC IP2IP Routing Table - Add Record Dialog Box



Add an entry and then configure it according to the table below.

Click the **Apply** button to save your changes.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

Table 18-6: IP2IP Routing Table Parameters

Parameter	Description
Matching Characteristics	
Source IP Group ID **[IP2IPRouting_SrcIPGroupID]**	Selects the IP Group from where the IP-to-IP call originated. Typically, the IP Group of an incoming SIP dialog is determined (or classified) using the 'Classification' table (see 'Configuring Classification Table' on page 178). If not used (i.e., any IP Group), simply leave the field empty. The default is -1.
Source Username Prefix **[IP2IPRouting_SrcUsernamePrefix]**	Defines the prefix of the user part of the incoming SIP dialog's source URI (usually the From URI). The default is "*". **Note:** The prefix can be a single digit or a range of digits. For available notations, see 'Dialing Plan Notation for Routing and Manipulation' on page 190.
Source Host **[IP2IPRouting_SrcHost]**	Defines the host part of the incoming SIP dialog's source URI (usually the From URI). If this rule is not required, leave the field empty. To denote any host name, use the asterisk (*) symbol. The default is "*".
Destination Username Prefix **[IP2IPRouting_DestUsernamePrefix]**	Defines the prefix of the incoming SIP dialog's destination URI (usually the Request URI) user part. If this rule is not required, leave the field empty. To denote any prefix, use the asterisk (*) symbol. The default is "*". **Note:** The prefix can be a single digit or a range of digits. For available notations, see 'Dialing Plan Notation for Routing and Manipulation' on page 190.
Destination Host **[IP2IPRouting_DestHost]**	Defines the host part of the incoming SIP dialog's destination URI (usually the Request-URI). If this rule is not required, leave the field empty. The asterisk (*) symbol can be used to denote any destination host. The default is "*".
Request Type **[IP2IPRouting_RequestType]**	Defines the SIP dialog request type of the incoming SIP dialog. **[0]** All (default) **[1]** INVITE **[2]** REGISTER **[3]** SUBSCRIBE **[4]** INVITE and REGISTER **[5]** INVITE and SUBSCRIBE **[6]** OPTIONS
Message Condition **[IP2IPRouting_MessageCondition]**	Selects a Message Condition rule. To configure Message Condition rules, see 'Configuring Condition Rules' on page 182.

Parameter	Description
Operation Routing Rule (when match occurs in characteristics)	
Destination Type **[IP2IPRouting_DestType]**	Determines the destination type to which the outgoing SIP dialog is sent. **[0]** IP Group (default) = The SIP dialog is sent to the IP Group's Proxy Set (SERVER-type IP Group) or registered contact from the database (if USER-type IP Group). **[1]** Dest Address = The SIP dialog is sent to the address configured in the following fields: 'Destination SRD ID', 'Destination Address', 'Destination Port', and 'Destination Transport Type'. **[2]** Request URI = The SIP dialog is sent to the address indicated in the incoming Request-URI. If the fields 'Destination Port' and 'Destination Transport Type' are configured, the incoming Request-URI parameters are overridden and these fields take precedence. **[3]** ENUM = An ENUM query is sent to include the destination address. If the fields 'Destination Port' and 'Destination Transport Type' are configured, the incoming Request-URI parameters are overridden and these fields take precedence. **[4]** Hunt Group = Used for call center survivability. For more information, see 'Call Survivability for Call Centers' on page 167.
Destination IP Group ID **[IP2IPRouting_DestIPGroupID]**	Defines the IP Group ID to where you want to route the call. The SIP dialog messages are sent to the IP address defined for the Proxy Set associated with this IP Group. If you select an IP Group, it is unnecessary to configure a destination IP address (in the 'Destination Address' field). However, if both parameters are configured, then the IP Group takes precedence. If the destination IP Group is of USER type, the device searches for a match between the Request-URI (of the received SIP dialog) to an AOR registration record in the device's database. The SIP dialog is then sent to the IP address of the registered contact. The default is -1. **Notes:** This parameter is only relevant if the parameter 'Destination Type' is set to 'IP Group'. However, regardless of the settings of the parameter 'Destination Type', the IP Group is still used - only for determining the IP Profile or outgoing SRD. If neither IP Group nor SRD are defined in this table, the destination SRD is determined according to the source SRD associated with the Source IP Group (configured in the IP Group table, see 'Configuring IP Groups' on page 114). If this table does not define an IP Group but only an SRD, then the first IP Group associated with this SRD (in the IP Group table) is used. If the selected destination IP Group ID is type SERVER, the request is routed according to the IP Group addresses. If the selected destination IP Group ID is type USER, the request is routed according to the IP Group specific database (i.e., only to registered users of the selected database). If the selected destination IP Group ID is ANY USER (**[-2**]), the request is routed according to the general database (i.e., any matching registered user).

Parameter	Description
Destination SRD ID **[IP2IPRouting_DestSRDID]**	Defines the SRD ID. The default is None. **Note:** The destination IP Group must belong to the destination SRD, if both are configured in this table.
Destination Address **[IP2IPRouting_DestAddress]**	Defines the destination IP address (or domain name, e.g., domain.com) to where the call is sent. **Notes:** This parameter is applicable only if the parameter 'Destination Type' is set to 'Dest Address' **[1].** When using domain names, enter a DNS server IP address or alternatively, define these names in the 'Internal DNS Table' (see 'Configuring the Internal SRV Table' on page 92).
Destination Port **[IP2IPRouting_DestPort]**	Defines the destination port to where the call is sent.
Destination Transport Type **[IP2IPRouting_DestTransportType]**	Defines the transport layer type for sending the call: **[-1]** Not Configured (default) **[0]** UDP **[1]** TCP **[2]** TLS **Note:** When this parameter is set to -1, the transport type is determined by the parameter SIPTransportType.
Alternative Route Options **[IP2IPRouting_AltRouteOptions]**	Determines whether this routing rule is the main routing rule or an alternative routing rule (to the rule defined directly above it in the table). **[0]** Route Row (default) = Main routing rule - the device first attempts to route the call to this route if the incoming SIP dialog's input characteristics matches this rule. **[1]** Alt Route Ignore Inputs = If the call cannot be routed to the main route (Route Row), the call is routed to this alternative route regardless of the incoming SIP dialog's input characteristics. **[2]** Alt Route Consider Inputs = If the call cannot be routed to the main route (Route Row), the call is routed to this alternative route only if the incoming SIP dialog matches this routing rule's input characteristics. **Notes:** The alternative routing entry (**[1]** or **[2]) must be defined in the next consecutive table entry index** to the Route Row entry (i.e., directly below it). For example, if Index 4 is configured as a Route Row, Index 5 must be configured as the alternative route. For IP-to-IP alternative routing, configure SBC alternative routing reasons upon receipt of 4xx, 5xx, and 6xx SIP responses (see 'Configuring Alternative Routing Reasons' on page 189). However, if no response, ICMP, or a SIP 408 response is received, the device attempts to use the alternative route even if no entries are configured in the 'SBC Alternative Routing Reasons' table. Multiple alternative route entries can be configured (e.g., Index 1 is the main route - Route Row - and indices 2 through 4 are configured as alternative routes).

Parameter	Description
Cost Group **[IP2IPRouting_CostGroup]**	Reserved for future use.

18.2.5.4 Configuring Alternative Routing Reasons

The SBC Alternative Routing Reasons page allows you to define up to five different call release (termination) reasons for call releases. If a call is released as a result of one of these reasons provided in SIP 4xx, 5xx, and 6xx response codes, the device attempts to locate an alternative route for the call. The call release reason type can be configured, for example, when there is no response to an INVITE message (after INVITE re-transmissions), where the device issues an internal 408 'No Response' implicit release reason.

Release reasons can also be configured to indicate that a route for an SRD or IP Group has reached its call admission control limit (i.e., maximum concurrent calls and/or call rate), as set in the Admission Control table (see 'Configuring Admission Control' on page 173). In such a scenario, an alternative route configured in the IP-to-IP Routing table can be used.

Alternative routes are configured in the IP2IP Routing table (see 'Configuring SBC IP-to-IP Routing Table' on page 183).

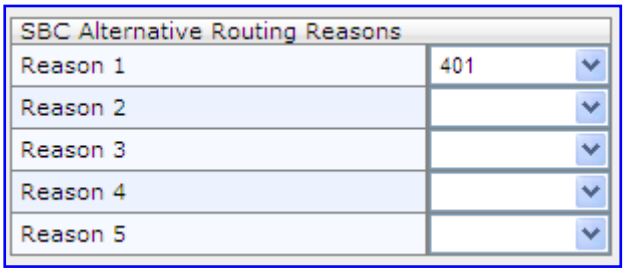
Notes:

- Alternative routing occurs even if this table is not configured upon scenarios where no response, ICMP, or a SIP 408 response is received.
- SIP requests pertaining to an SRD or IP Group that reach the call limit (maximum concurrent calls and/or call rate) as defined in the Call Admission table are sent to an alternative route if configured in the IP2IP Routing table for the SRD or IP Group. If no alternative routing rule is located, the device automatically rejects the SIP request with a SIP 486 "Busy Here" response.
- You can also configure alternative routing reasons using the *ini* file table parameter SBCAlternativeRoutingReasons.

To configure SIP reason codes for alternative IP routing:

1. Open the SBC Alternative Routing Reasons page (**Configuration** tab >**VoIP** menu >**SBC** submenu >**Routing SBC** submenu >**Alternative Routing Reasons**).

Figure 18-24: Alternative Routing Reasons Page



Configure different call failure reasons that invoke alternative routing.

Click Subm**it to** apply your changes.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

18.2.6 Dialing Plan Notation for Routing and Manipulation

The device supports flexible dialing plan notations for denoting the prefix and/or suffix source and/or destination numbers and SIP URI user names in the routing and manipulation tables.

Table 18-7: Dialing Plan Notations for Prefixes and Suffixes

Notation	Description
x (letter "x")	Denotes any single digit.
# (pound symbol)	When used at the end of a prefix, it denotes the end of a number. For example, **54324xx#** represents a 7-digit number that starts with the digits 54324. When used anywhere in the suffix, it is part of the number. For example, **(3#45)** can represent the number string, 123#45.
***** (asterisk symbol)	When used in the prefix, it denotes any number. When used in the suffix, it is part of the number. For example, **(3*45)** can represent the number string, 123*45.
Range of Digits **Notes:** Dial plans denoting a prefix that is a range must be enclosed in square brackets, e.g., **[4-8]** or **23xx[456]**. Dial plans denoting a prefix that is not a range is not enclosed, e.g., **12345#**. Dial plans denoting a suffix must be enclosed in parenthesis, e.g., **(4)** and **(4-8)**. Dial plans denoting a suffix that include multiple ranges, the range must be enclosed in square brackets, e.g., **(23xx[4,5,6])**. An example for entering a combined prefix and suffix dial plan - assume you want to match a rule whose destination phone prefix is 4 to 8, and suffix is 234, 235, or 236. The entered value would be the following: **[4-8](23[4,5,6])**.	
[n-m] or **(n-m)**	Represents a range of numbers. For example: To denote numbers from 5551200 to 5551300: Prefix: **[5551200-5551300]#** Suffix: **(5551200-5551300)** To denote numbers from 123100 to 123200: Prefix: **123[100-200]** Suffix: **(123[100-200])** To denote prefix and suffix numbers together: 03(100): for any number that starts with 03 and ends with 100. **[100-**199](100,101,105): for a number that starts with 100 to 199 and ends with 100, 101 or 105. 03(abc): for any number that starts with 03 and ends with abc. 03(5xx): for any number that starts with 03 and ends with 5xx. 03(400,401,405): for any number that starts with 03 and ends with 400 or 401 or 405. **Notes:** The value *n* must be less than the value *m*. Only numerical ranges are supported (not alphabetical letters). For suffix ranges, the starting (*n*) and ending (*m*) numbers in the range must have the same number of digits. For example, (23-34) is correct, but (3-12) is not.

Notation	Description
[n,m,...] or **(n,m,...)**	Represents multiple numbers. For example, to denote a one-digit number starting with 2, 3, 4, 5, or 6: Prefix: **[2,3,4,5,6]#** Suffix: **(2,3,4,5,6)** Prefix with Suffix: **[2,3,4,5,6](8,7,6)** - prefix is denoted in square brackets; suffix in parenthesis For **prefix only**, the notations *d[n,m]e* and *d[n-m]e* can also be used: To denote a five-digit number that starts with 11, 22, or 33: **[11,22,33]xxx#** To denote a six-digit number that starts with 111 or 222: **[111,222]xxx#** **Note:** Up to three digits can be used to denote each number.
[n1-m1,n2-m2,a,b,c,n3-m3] or **(n1-m1,n2-m2,a,b,c,n3-m3)**	Represents a mixed notation of single numbers and multiple ranges. For example, to denote numbers 123 to 130, 455, 766, and 780 to 790: Prefix: **[123-130,455,766,780-790]** Suffix: **(123-130,455,766,780-790)** **Note:** The ranges and the single numbers used in the dial plan must have the same number of digits. For example, each number range and single number in the dialing plan example above consists of three digits.

18.2.7 Manipulations SBC

This section describes the configuration of the manipulation rules for the SBC application. These include the following:

SIP message manipulation - see 'SIP Message Manipulations' on page 192

IP-to-IP inbound manipulation - see 'Configuring IP-to-IP Inbound Manipulations' on page 195

IP-to-IP outbound manipulation - see 'Configuring IP-to-IP Outbound Manipulations' on page 198

18.2.7.1 Configuring Message Manipulations

The Message Manipulations page allows you to define up to 200 SIP message manipulation rules. This manipulation includes insertion, removal, and/or modification of SIP headers. Multiple manipulation rules can be configured for the same SIP message.

SIP message manipulation rules are assigned to IP Groups in the IP Group table and can be applied to inbound or outbound messages.

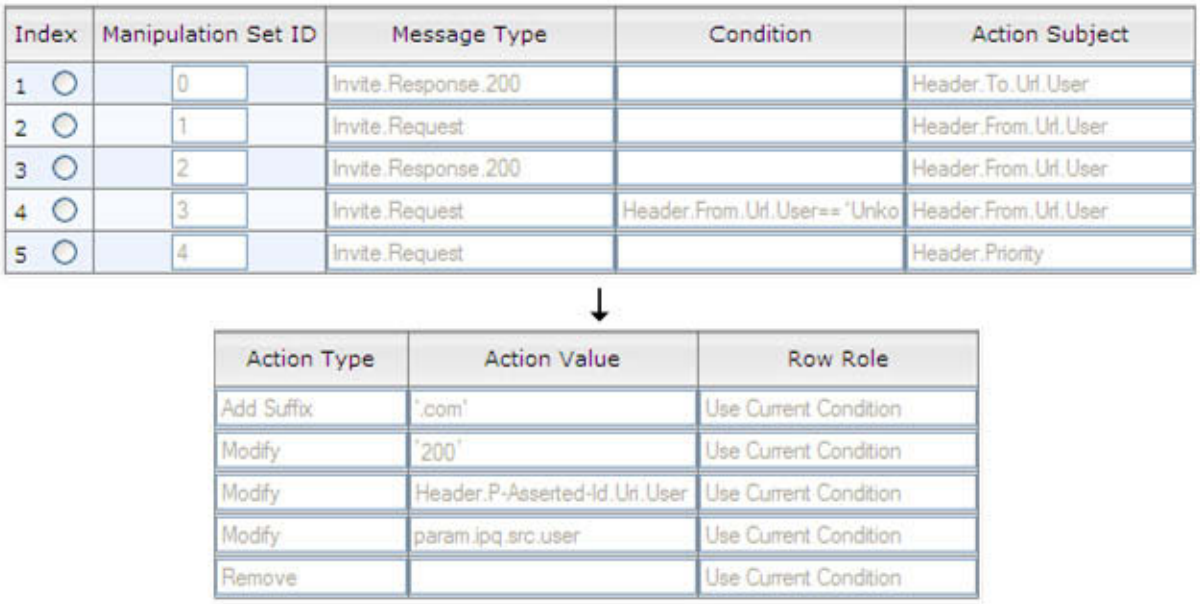
Notes:

- For more information on the syntax for configuring SIP message manipulation rules in the Message Manipulation table, see Appendix B, 'SIP Message Manipulation Description.
- The values entered in the table are not case-sensitive.
- Each message can be manipulated twice - once for the source leg manipulation rules and once in the destination leg (source and destination IP Groups).
- Unknown SIP parts can only be added or removed.
- SIP manipulations do not allow you to remove or add mandatory SIP headers. They can only be modified and only on requests that initiate new dialogs. Mandatory SIP headers include To, From, Via, CSeq, Call-Id, and Max-Forwards.
- Manipulation of SDP body is currently not supported.
- For configuring Message Manipulation using the *ini* file, see the parameter MessageManipulations.

To configure SIP message manipulation rules:

1. Open the Message Manipulations page (**Configuration** tab >**VoIP** menu >**SBC** submenu >**Manipulations SBC** submenu >**Message**).

Figure 18-25: Message Manipulations Page



Index	Manipulation Set ID	Message Type	Condition	Action Subject
1	0	Invite.Response.200		Header.To.Url.User
2	1	Invite.Request		Header.From.Url.User
3	2	Invite.Response.200		Header.From.Url.User
4	3	Invite.Request	Header.From.Url.User== 'Unko	Header.From.Url.User
5	4	Invite.Request		Header.Priority

Action Type	Action Value	Row Role
Add Suffix	'.com'	Use Current Condition
Modify	'200'	Use Current Condition
Modify	Header.P-Asserted-Id.Url.User	Use Current Condition
Modify	param.ipq.src.user	Use Current Condition
Remove		Use Current Condition

The previous figure shows the following message manipulation rules:

Index 1: adds the suffix ".com" to the host part of the To header.

Index 2: changes the user part of the SIP From header to 200.

Index 3: changes the user part of the From header to the user part of the P-Asserted-ID.

Index 4: if the user part of the From header equals "unknown", then it is changed according to the srcIPGroup call's parameter.

Index 5: removes the Priority header from an incoming INVITE message.

Add an entry and then configure it according to the table below.

Click the **Apply** button to save your changes.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

Table 18-8: Message Manipulations Parameters

Parameter	Description
Manipulation Set ID **[ManSetID]**	Defines a Manipulation Set ID for the rule. You can define the same Manipulation Set ID for multiple rules and thereby, create a group of rules that you can assign to an IP entity. The Manipulation Set IDs are later used to assign the manipulation rules to an IP Group (see 'Configuring IP Groups' on page 114) for inbound and/or outbound messages.
Matching Characteristics	
Message Type **[MessageType]**	Defines the SIP message type that you want to manipulate. The valid value is a string denoting the SIP message. For example: Empty = rule applies to all messages Invite = rule applies to all INVITE requests and responses Invite.Request = rule applies to INVITE requests Invite.Response = rule applies to INVITE responses subscribe.response.2xx = rule applies to SUBSCRIBE confirmation responses **Note:** Currently, SIP 100 Trying messages cannot be manipulated.
Condition **[Condition]**	Defines the condition that must exist for the rule to apply. The valid value is a string. For example: header.from.url.user== '100' (indicates that the user part of the From header must have the value "100") header.contact.param.expires >'3600' header.to.url.host contains 'domain' param.call.dst.user != '100' **Note:** Currently, SDP body message types are not supported.
Operation	
Action Subject **[ActionSubject]**	Defines the SIP header upon which the manipulation is performed.
Action Type **[ActionType]**	Defines the type of manipulation. **[0]** Add (default) = adds new header/param/body (header or parameter elements). **[1]** Remove = removes header/param/body (header or parameter elements). **[2]** Modify = sets element to the new value (all element types). **[3]** Add Prefix = adds value at the beginning of the string (string element only). **[4]** Add Suffix = adds value at the end of the string (string element only). **[5]** Remove Suffix = removes value from the end of the string (string element only). **[6]** Remove Prefix = removes value from the beginning of the string (string element only).

Parameter	Description
Action Value **[ActionValue]**	Defines a value (string) that you want to use in the manipulation. The syntax is as follows: string/<message-element>/<call-param> + string/<message-element>/<call-param> For example: 'itsp.com' header.from.url.user param.call.dst.user param.call.dst.host + '.com' param.call.src.user + '<' + header.from.url.user + '@' + header.p-asserted-id.url.host + '>' **Note:** Only single quotation marks must be used.
Row Role **[RowRole]**	Determines which condition must be used for the rule of this table row. **[0]** Use Current Condition = The condition entered in this row must be matched in order to perform the defined action (default). **[1]** Use Previous Condition = The condition of the rule configured directly above this row must be used in order to perform the defined action. This option allows you to configure multiple actions for the same condition. **Note:** When multiple manipulations rules apply to the same header, the next rule applies to the result string of the previous rule.

18.2.7.2 Configuring IP-to-IP Inbound Manipulations

The IP to IP Inbound Manipulation page allows you to configure up to 100 manipulation rules for manipulating the SIP URI user part (source and destination) of inbound SIP dialog requests. You can apply these manipulations to different SIP dialog message types (e.g., INVITE or REGISTER).

Manipulated destination URI user part are done on the following SIP headers: Request-URI, To, and Remote-Party-ID (if exists)

Manipulated source URI user part are done on the following SIP headers: From, P-Asserted (if exists), P-Preferred (if exists), and Remote-Party-ID (if exists)

Notes:

For a specific manipulation rule to be effective, the incoming SIP dialog must match the configured characteristics.

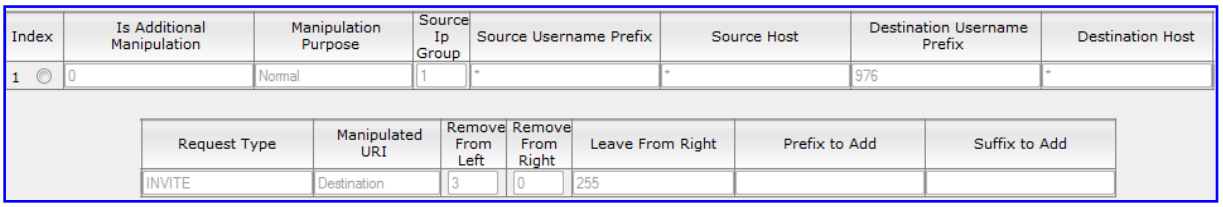
SIP URI host name (source and destination) manipulations are configured in the IP Group table (see 'Configuring IP Groups' on page 114). These manipulations are simply host name substitutions with the names defined for the source and destination IP Groups respectively.

The IP to IP Inbound Manipulation table can also be configured using the *ini* file table parameter IPInboundManipulation (see 'SBC Parameters' on page 339).

To configure IP-to-IP inbound manipulation rules:

1. Open the IP to IP Inbound Manipulation page (**Configuration** tab >**VoIP** menu >**SBC** submenu >**Manipulations SBC** submenu >**IP to IP Inbound**).

Figure 18-26: IP to IP Inbound Manipulation Page



The figure above shows a manipulation configuration example that removes the destination URI user name prefix "976" in incoming INVITE messages received from IP Group #1 .

Add an entry and then configure it according to the table below.

Click the **Apply** button to save your changes.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

Table 18-9: IP to IP Inbound Manipulation Parameters

Parameter	Description
Matching Characteristics	
Is Additional Manipulation **[IsAdditionalManipulation]**	Determines whether additional SIP URI user part manipulation is done for the table entry rule listed directly above it. **[0]** 0 = Regular manipulation rule (not done in addition to the rule above it). **[1]** 1 = If the above row entry rule matched the call, consider this row entry as a match as well and perform the manipulation specified by this rule. **Note:** Additional manipulation can only be performed on a different SIP URI (either source or destination) to the rule configured in the row above (defined by the parameter ManipulatedURI).
Manipulation Purpose **[ManipulationPurpose]**	Defines the purpose of the manipulation: **[0]** Normal = Inbound manipulations affect the routing input and source and/or destination number (default). **[1]** Routing input only = Inbound manipulations affect the routing input only, retaining the original source and destination number. **[2]** Shared Line = Used for the Shared-Line Appearance feature. This manipulation is for registration requests to change the destination number of the secondary extension numbers to the primary extension.
Source IP Group **[SrcIpGroup]**	Defines the IP Group from where the incoming INVITE is received. For any Source IP Group, enter the value -1.
Source Username Prefix **[SrcUsernamePrefix]**	Defines the prefix of the source SIP URI user name (usually in the From header). For any prefix, enter the asterisk "*" symbol (default). **Note:** The prefix can be a single digit or a range of digits. For available notations, see 'Dialing Plan Notation for Routing and Manipulation' on page 190.
Source Host **[SrcHost]**	Defines the source SIP URI host name - full name (usually in the From header). For any host name, enter the asterisk "*" symbol (default).
Destination Username Prefix **[DestUsernamePrefix]**	Defines the prefix of the destination SIP URI user name (usually in the Request-URI). For any prefix, enter the asterisk "*" symbol (default). **Note:** The prefix can be a single digit or a range of digits. For available notations, see 'Dialing Plan Notation for Routing and Manipulation' on page

Parameter	Description
	190.
Destination Host **[DestHost]**	Defines the destination SIP URI host name - full name (usually in the Request URI). For any host name, enter the asterisk "*" symbol (default).
Request Type **[RequestType]**	Defines the SIP request type to which the manipulation rule is applied. **[0]** All = all SIP messages (default) **[1]** INVITE = all SIP messages except REGISTER and SUBSCRIBE **[2]** REGISTER = only SIP REGISTER messages **[3]** SUBSCRIBE = only SIP SUBSCRIBE messages **[4]** INVITE and REGISTER = all SIP messages except SUBSCRIBE **[5]** INVITE and SUBSCRIBE = all SIP messages except REGISTER
Manipulated URI **[ManipulatedURI]**	Determines whether the source or destination SIP URI user part is manipulated. **[0]** Source = Manipulation is done on the source SIP URI user part. (default) **[1]** Destination = Manipulation is done on the destination SIP URI user part.
Operation Manipulation Rule (when match occurs in characteristics)	
Remove From Left **[RemoveFromLeft]**	Defines the number of digits to remove from the left of the user name prefix. For example, if you enter 3 and the user name is "john", the new user name is "n".
Remove From Right **[RemoveFromRight]**	Defines the number of digits to remove from the right of the user name prefix. For example, if you enter 3 and the user name is "john", the new user name is "j".
Leave From Right **[LeaveFromRight]**	Defines the number of characters that you want retained from the right of the user name.
Prefix to Add **[Prefix2Add]**	Defines the number or string that you want added to the front of the user name. For example, if you enter 'user' and the user name is "john", the new user name is "userjohn".
Suffix to Add **[Suffix2Add]**	Defines the number or string that you want added to the end of the user name. For example, if you enter '01' and the user name is "john", the new user name is "john01".

18.2.7.3 Configuring IP-to-IP Outbound Manipulations

The IP to IP Outbound Manipulation page allows you to configure up to 100 manipulation rules for manipulating SIP URI user part (source and destination) of outbound SIP dialog requests. Manipulation rules in the table are located according to the source IP Group, and source and destination host and user prefixes and can be applied to a user-defined SIP request type (e.g., INVITE, OPTIONS, SUBSCRIBE, and /or REGISTER). However, since outbound manipulations are done only after routing, the outbound manipulation rule matching can also be done by destination IP Group.

Manipulated destination URI user part are performed on the following SIP headers: Request URI, To, and Remote-Party-ID (if exists). Manipulated source URI user part are performed on the following SIP headers: From, P-Asserted (if exists), P-Preferred (if exists), and Remote-Party-ID (if exists).

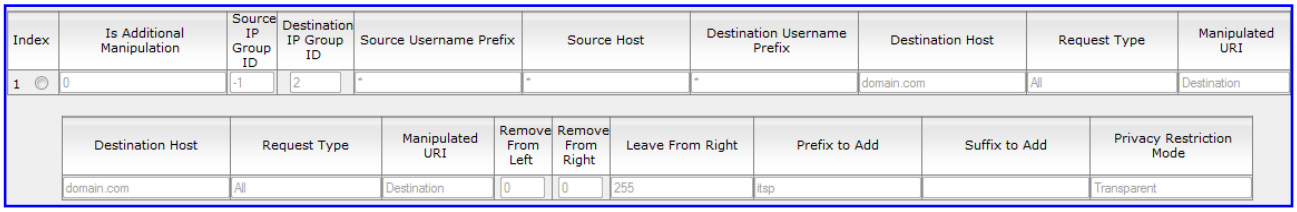
Notes:

- For a specific manipulation rule to be effective, the incoming SIP dialog must match the characteristics configured for that rule.
- SIP URI host name (source and destination) manipulations are configured in the IP Group table. These manipulations are simply host name substitutions with the names defined for the source and destination IP Groups respectively.
- The IP to IP Outbound Manipulation table can also be configured using the *ini* file table parameter IPOutboundManipulation (see 'SBC Parameters' on page 339).

To configure IP-to-IP outbound manipulation rules:

1. Open the IP to IP Outbound Manipulation page (**Configuration** tab >**VoIP** menu >**SBC** submenu >**Manipulations SBC** submenu >**IP to IP Outbound**).

Figure 18-27: IP to IP Outbound Manipulation Page



Add an entry and then configure it according to the table below.

Click the **Apply** button to save your changes.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

Table 18-10: IP to IP Outbound Manipulation Table Parameters

Parameter	Description
Matching Characteristics	
Is Additional Manipulation **[IsAdditionalManipulation]**	Determines whether additional SIP URI user part manipulation is done for the table entry rule listed directly above it. **[0]** 0 = Regular manipulation rule - not done in addition to the rule above it (default). **[1]** 1 = If the previous table row entry rule matched the call, consider this row entry as a match as well and perform the manipulation specified by this rule. **Note:** Additional manipulation can only be performed on a different SIP URI

Parameter	Description
	(either source or destination) to the rule configured in the row above (defined by the parameter ManipulatedURI).
Source IP Group ID **[SrcIPGroupID]**	Defines the IP Group from where the INVITE is received. For any Source IP Group, enter the value -1.
Destination IP Group ID **[DestIPGroupID]**	Defines the IP Group to where the INVITE is to be sent. For any Destination IP Group, enter the value -1.
Source Username Prefix **[SrcUsernamePrefix]**	Defines the prefix of the source SIP URI user name (usually in the From header). For any prefix, enter the asterisk "*" symbol (default). **Note:** The prefix can be a single digit or a range of digits. For available notations, see 'Dialing Plan Notation for Routing and Manipulation' on page 190.
Source Host **[SrcHost]**	Defines the source SIP URI host name - full name (usually in the From header). For any host name, enter the asterisk "*" symbol (default).
Destination Username Prefix **[DestUsernamePrefix]**	Defines the prefix of the destination SIP URI user name (usually in the Request-URI). For any prefix, enter the asterisk "*" symbol (default). **Note:** The prefix can be a single digit or a range of digits. For available notations, see 'Dialing Plan Notation for Routing and Manipulation' on page 190.
Destination Host **[DestHost]**	Defines the destination SIP URI host name - full name (usually in the Request URI). For any host name, enter the asterisk "*" symbol (default).
Request Type **[RequestType]**	Defines the SIP request type to which the manipulation rule is applied. **[0]** All = all SIP messages (default) **[1]** INVITE = all SIP messages except REGISTER and SUBSCRIBE **[2]** REGISTER = only SIP REGISTER messages **[3]** SUBSCRIBE = only SIP SUBSCRIBE messages **[4]** INVITE and REGISTER = all SIP messages except SUBSCRIBE **[5]** INVITE and SUBSCRIBE = all SIP messages except REGISTER
Manipulated URI **[IsAdditionalManipulation]**	Determines whether the source or destination SIP URI user part is manipulated. **[0]** Source = Manipulation is done on the source SIP URI user part (default). **[1]** Destination = Manipulation is done on the destination SIP URI user part.
Operation Manipulation Rule (when match occurs in characteristics)	
Remove From Left **[RemoveFromLeft]**	Defines the number of digits to remove from the left of the user name prefix. For example, if you enter 3 and the user name is "john", the new user name is "n".
Remove From Right **[RemoveFromRight]**	Defines the number of digits to remove from the right of the user name prefix. For example, if you enter 3 and the user name is "john", the new user name is "j".
Leave From Right **[LeaveFromRight]**	Defines the number of characters that you want retained from the right of the user name.
Prefix to Add **[Prefix2Add]**	Defines the number or string that you want added to the front of the user name. For example, if you enter 'user' and the user name is "john", the new user name is "userjohn".
Suffix to Add **[Suffix2Add]**	Defines the number or string that you want added to the end of the user name. For example, if you enter '01' and the user name is "john", the new user name is "john01".

Parameter	Description
Privacy Restriction Mode **[PrivacyRestrictionMode]**	Determines user privacy handling (i.e., restricting source user identity in outgoing SIP dialogs). **[0]** Transparent = No intervention in SIP privacy (default). **[1]** Don't change privacy = The user identity remains the same as in the incoming SIP dialog. If a restricted number exists, the restricted presentation is normalized as follows: From URL header: anonymous@anonymous.invalid. If a P-Asserted-Identity header exists (either in the incoming SIP dialog or added by the device), a Privacy header is added with the value "id". **[2]** Restrict = The user identity is restricted (the restricted presentation is as mentioned above). **[3]** Remove Restriction = The device attempts to reveal the user identity by setting user values in the From header and removing the privacy "id" value if the Privacy header exists. If the From header user is anonymous, the value is taken from the P-Preferred-Identity, P-Asserted-Identity, or Remote-Party-ID header (if exists). The device identifies an incoming user as restricted if one of the following exists: From header user is anonymous. P-Asserted-Identity and Privacy headers contain the value "id". **Note:** All restriction logic is performed after the user number has been manipulated.

19 Stand-Alone Survivability (SAS) Application

This section describes the Sand-Alone Survivability application.

19.1 Overview

The device's Stand-Alone Survivability (SAS) feature ensures telephony communication continuity (survivability) for enterprises using hosted IP services (such as IP Centrex) or IP-PBX in cases of failure of these entities. In case of failure of the IP Centrex, IP-PBX servers (or even WAN connection and access Internet modem), the enterprise typically loses its internal telephony service at any branch, between its offices, and with the external environment.

Notes:

- The SAS application is available only if the device is installed with the SAS Software Upgrade Key.
- Throughput this section, the term *user agent* (UA) refers to the enterprise's LAN phone user (i.e., SIP telephony entities such as IP phones).
- Throughout this section, the term *proxy* or *proxy server* refers to the enterprise's centralized IP Centrex or IP-PBX.
- Throughout this section. the term *SAS* refers to the SAS application running on the device.

19.1.1 SAS Operating Modes

The device's SAS application can be implemented in one of the following main modes:

Outbound Proxy: In this mode, SAS receives SIP REGISTER requests from the enterprise's UAs and forwards these requests to the external proxy (i.e., outbound proxy). When a connection with the external proxy fails, SAS enters SAS emergency state and serves as a proxy, by handling internal call routing for the enterprise's UAs - routing calls between UAs. For more information, see 'SAS Outbound Mode' on page 202.

Redundant Proxy: In this mode, the enterprise's UAs register with the external proxy and establish calls directly through the external proxy, without traversing SAS (or the device per se'). Only when connection with the proxy fails, do the UAs register with SAS, serving now as the UAs redundant proxy. SAS then handles the calls between UAs. This mode is operational only during SAS in emergency state. This mode can be implemented, for example, for proxies that accept only SIP messages that are sent directly from the UAs. For more information, see 'SAS Redundant Mode' on page 204.

Note: It is recommended to implement the SAS outbound mode.

19.1.1.1 SAS Outbound Mode

This section describes the SAS outbound mode, which includes the following states:

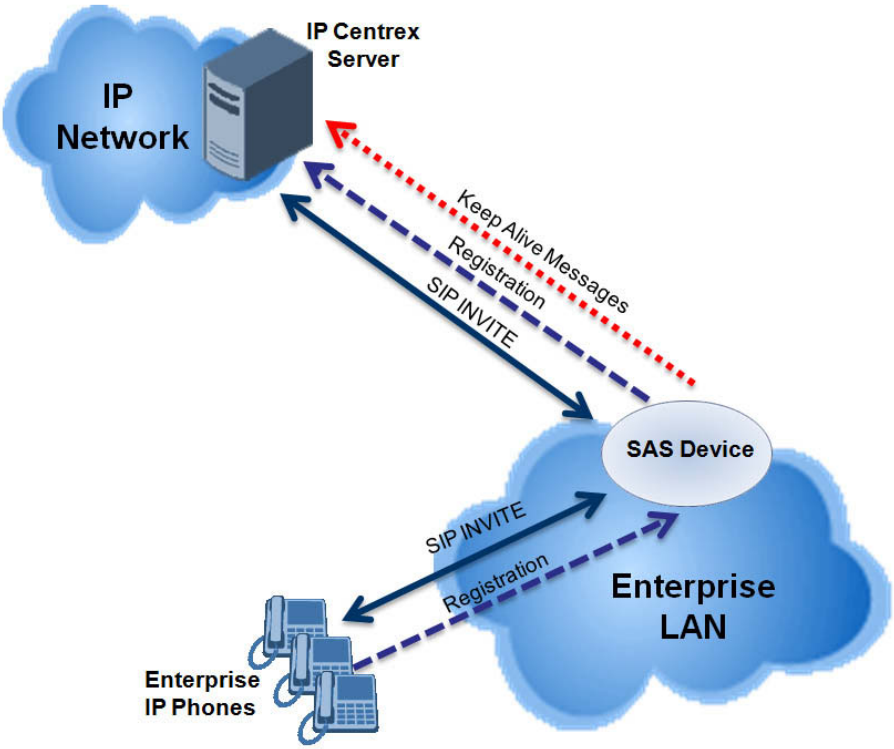
Normal state (see 'Normal State' on page 202)

Emergency state (see 'Emergency State' on page 203)

19.1.1.1.1 Normal State

In normal state, SAS receives REGISTER requests from the enterprise's UAs and forwards them to the external proxy (i.e., outbound proxy). Once the proxy replies with a SIP 200 OK, the device records the Contact and address of record (AOR) of the UAs in its internal SAS registration database. Therefore, in this mode, SAS maintains a database of all the registered UAs in the network. In addition, SAS continuously maintains a keep-alive mechanism toward the external proxy, using SIP OPTIONS messages. The figure below illustrates the operation of SAS outbound mode in normal state:

Figure 19-1: SAS Outbound Mode in Normal State (Example)



19.1.1.1.2 Emergency State

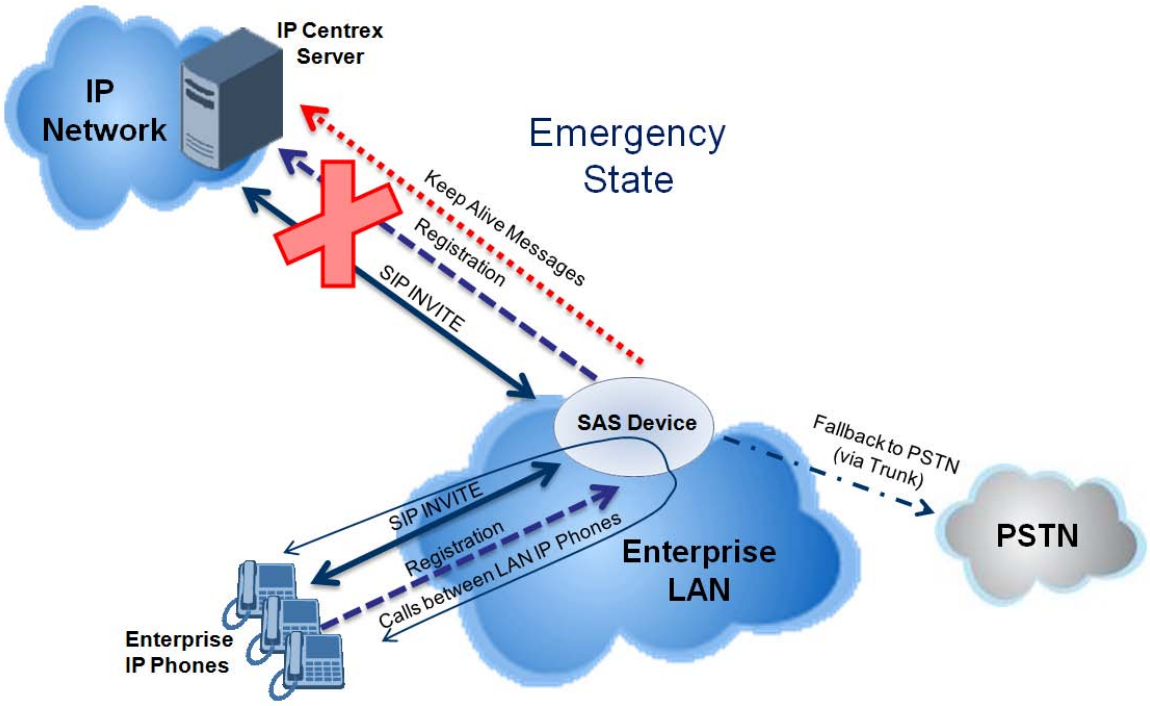
When a connection with the external proxy fails (detected by the device's keep-alive messages), the device enters SAS emergency state. The device serves as a proxy for the UAs, by handling internal call routing of the UAs (within the LAN enterprise).

When the device receives calls, it searches its SAS registration database to locate the destination address (according to AOR or Contact). If the destination address is not found, SAS forwards the call to the default gateway. Typically, the default gateway is defined as the device itself (on which SAS is running).

The routing logic of SAS in emergency state is described in detail in 'SAS Routing in Emergency State' on page 208.

The figure below illustrates the operation of SAS outbound mode in emergency state:

Figure 19-2: SAS Outbound Mode in Emergency State (Example)



When emergency state is active, SAS continuously attempts to communicate with the external proxy, using keep-alive SIP OPTIONS. Once connection to the proxy returns, the device exits SAS emergency state and returns to SAS normal state, as explained in 'Exiting Emergency and Returning to Normal State' on page 205.

19.1.1.2 SAS Redundant Mode

In SAS redundant mode, the enterprise's UAs register with the external proxy and establish calls directly through it, without traversing SAS (or the device per se'). Only when connection with the proxy fails, do the UAs register with SAS, serving now as the UAs redundant proxy. SAS then handles the calls between UAs, and between the UAs.

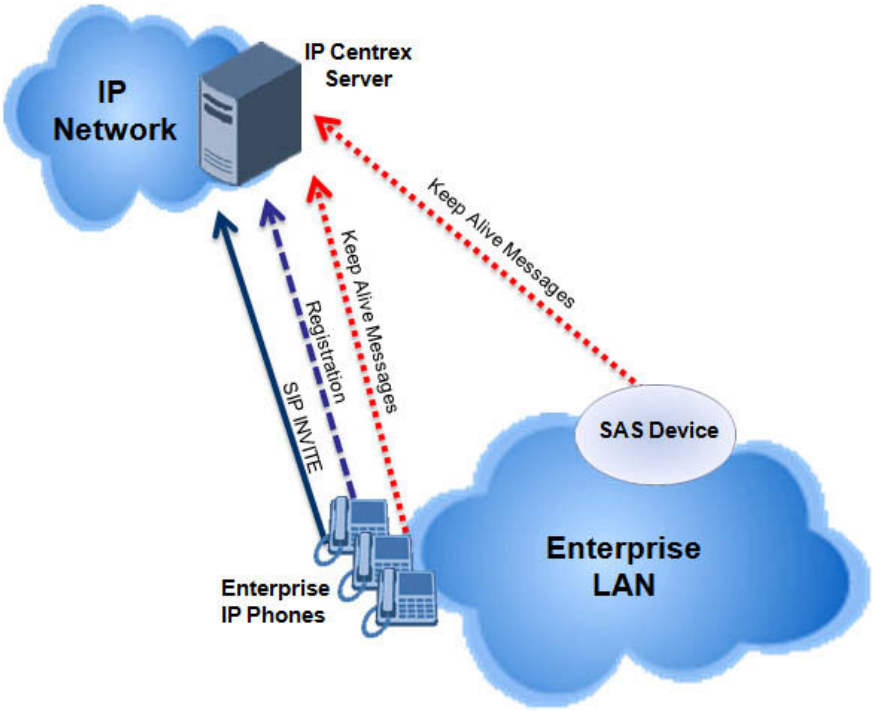
This mode is operational only during SAS in emergency state.

Note: In this SAS deployment, the UAs (e.g., IP phones) must support configuration for primary and secondary proxy servers (i.e., proxy redundancy), as well as homing. Homing allows the UAs to switch back to the primary server from the secondary proxy once the connection to the primary server returns (UAs check this using keep-alive messages to the primary server). If homing is not supported by the UAs, you can configure SAS to ignore messages received from UAs in normal state (the 'SAS Survivability Mode' parameter must be set to 'Always Emergency' / 2) and thereby, “force” the UAs to switch back to their primary proxy.

19.1.1.2.1 Normal State

In normal state, the UAs register and operate directly with the external proxy.

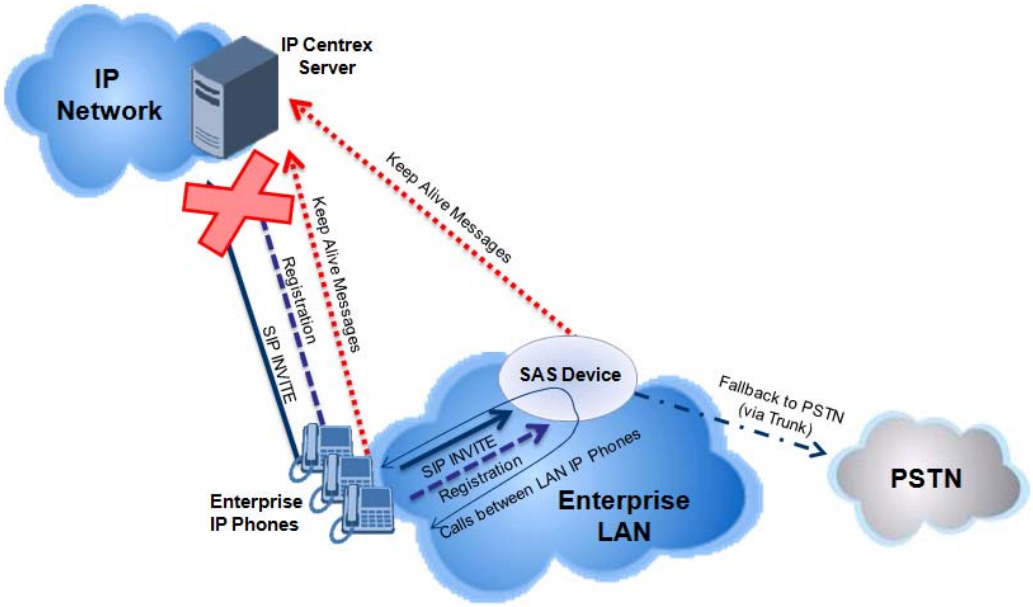
Figure 19-3: SAS Redundant Mode in Normal State (Example)



19.1.1.2.2Emergency State

If the UAs detect that their primary (external) proxy does not respond, they immediately register to SAS and start routing calls to it.

Figure 19-4: SAS Redundant Mode in Emergency State (Example)



19.1.1.2.3Exiting Emergency and Returning to Normal State

Once the connection with the primary proxy is re-established, the following occurs:

UAs: switch back to operate with the primary proxy.

SAS: ignores REGISTER requests from the UAs, forcing the UAs to switch back to the primary proxy.

Note: This is applicable only if the 'SAS Survivability Mode' parameter is set to 'Always Emergency' (2).

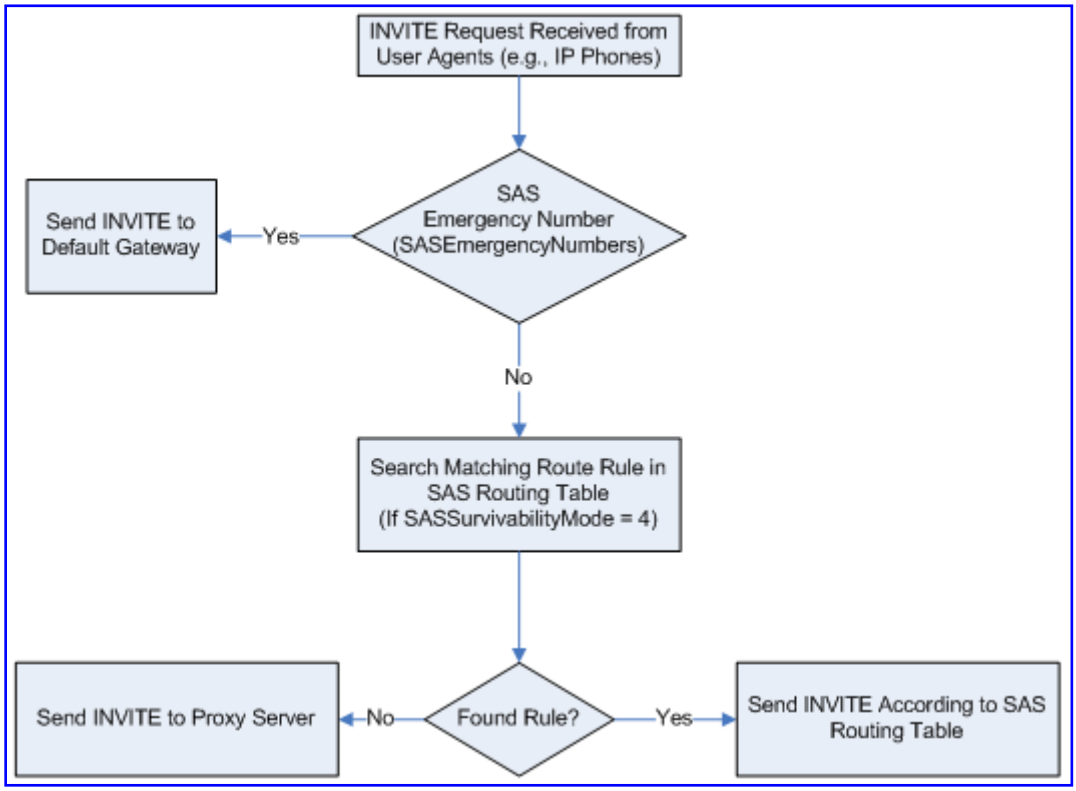
19.1.2 SAS Routing

This section provides flowcharts describing the routing logic for SAS in normal and emergency states.

19.1.2.1 SAS Routing in Normal State

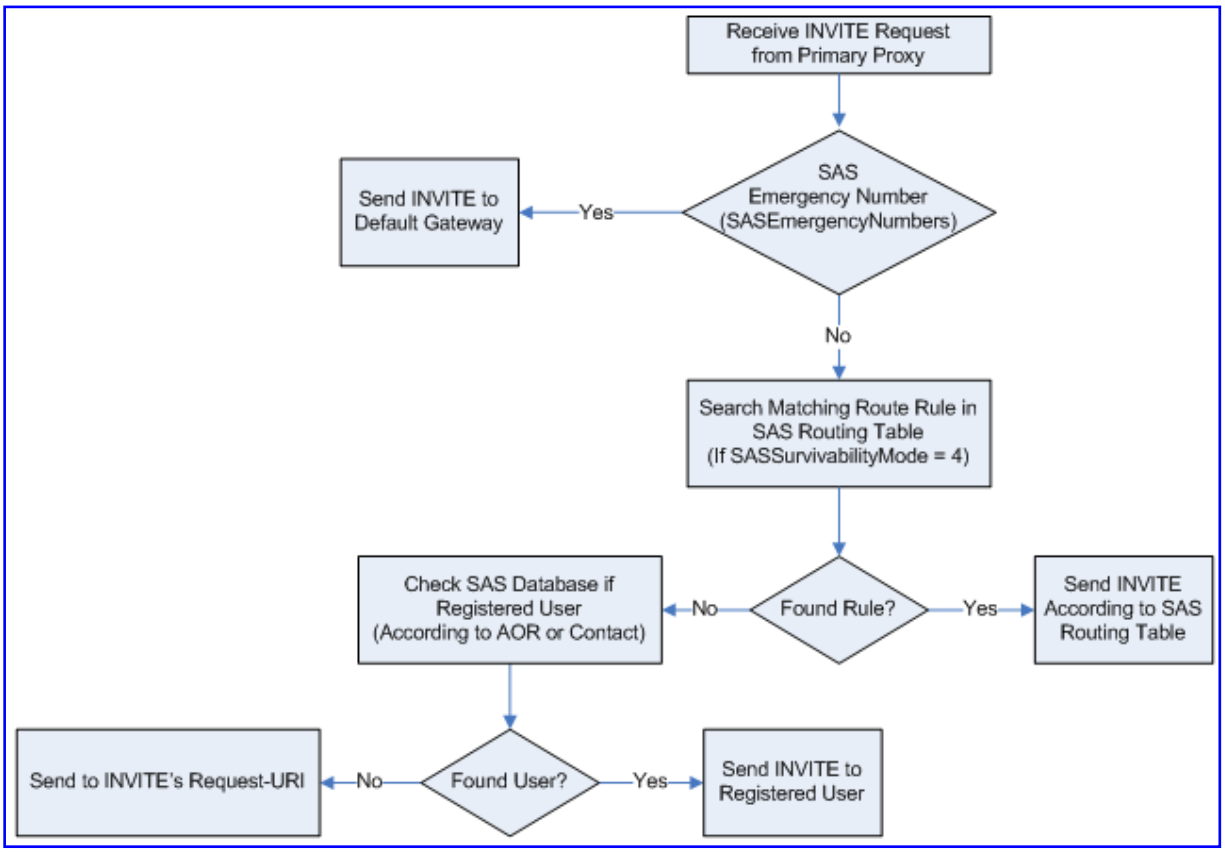
The flowchart below displays the routing logic for SAS in normal state for INVITE messages received from the UAs:

Figure 19-5: Flowchart of INVITE from UA's in SAS Normal State


INVITE Request Received from User Agents (e.g., IP Phones)
SAS Emergency Number (SASEmergencyNumbers)
Yes
Send INVITE to Default Gateway
No
Search Matching Route Rule in SAS Routing Table (If SASSurvivabilityMode = 4)
Found Rule?
No
Send INVITE to Proxy Server
Yes
Send INVITE According to SAS Routing Table

The flowchart below displays the routing logic for SAS in normal state for INVITE messages received from the external proxy:

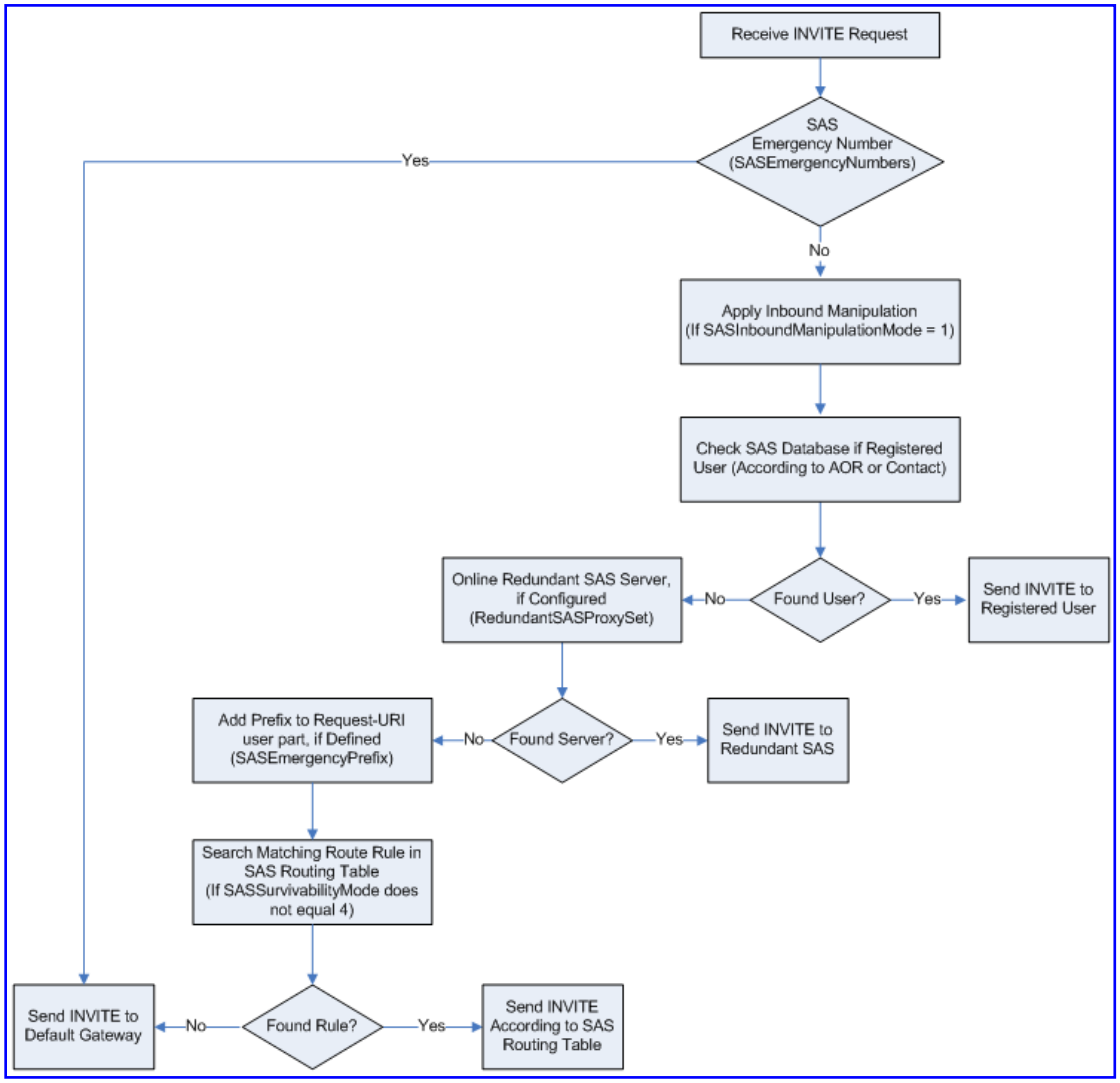
Figure 19-6: Flowchart of INVITE from Primary Proxy in SAS Normal State


Receive INVITE Request from Primary Proxy
SAS Emergency Number (SASEmergencyNumbers)
Yes
Send INVITE to Default Gateway
No
Search Matching Route Rule in SAS Routing Table (If SASSurvivabilityMode = 4)
Found Rule?
No
Check SAS Database if Registered User (According to AOR or Contact)
Yes
Send INVITE According to SAS Routing Table
Found User?
No
Send to INVITE's Request-URI
Yes
Send INVITE to Registered User

19.1.2.2 SAS Routing in Emergency State

The flowchart below shows the routing logic for SAS in emergency state:

Figure 19-7: Flowchart for SAS Emergency State


Receive INVITE Request
SAS Emergency Number (SASEmergencyNumbers)
Yes
No
Apply Inbound Manipulation (If SASInboundManipulationMode = 1)
Check SAS Database if Registered User (According to AOR or Contact)
Found User?
No
Yes
Send INVITE to Registered User
Online Redundant SAS Server, if Configured (RedundantSASProxySet)
Found Server?
No
Yes
Send INVITE to Redundant SAS
Add Prefix to Request-URI user part, if Defined (SASEmergencyPrefix)
Search Matching Route Rule in SAS Routing Table (If SASSurvivabilityMode does not equal 4)
Found Rule?
No
Yes
Send INVITE to Default Gateway
Send INVITE According to SAS Routing Table

19.2 SAS Configuration

SAS supports various configuration possibilities, depending on how the device is deployed in the network and the network architecture requirements. This section provides step-by-step procedures on configuring the SAS application, using the device's Web interface.

The SAS configuration includes the following:

General SAS configuration that is common to all SAS deployment types (see 'General SAS Configuration' on page 209)

SAS outbound mode (see 'Configuring SAS Outbound Mode' on page 213)

SAS redundant mode (see 'Configuring SAS Redundant Mode' on page 214)

Optional, advanced SAS features (see 'Advanced SAS Configuration' on page 215)

19.2.1 General SAS Configuration

This section describes the general configuration required for the SAS application. This configuration is applicable to all SAS modes.

19.2.1.1 Enabling the SAS Application

Before you can configure SAS, you need to enable the SAS application on the device. Once enabled, the device's Web interface provides the SAS pages for configuring SAS.

Note: The SAS application is available only if the device is installed with the SAS Software Upgrade Key. If your device is not installed with the SAS feature, contact your representative.

To enable the SAS application:

1. Open the Applications Enabling page (**Configuration** tab >**VoIP** menu >**Applications Enabling** > **Applications Enabling**).

From the 'SAS Application' drop-down list, select **Enable**.

Figure 19-8: Enabling SAS

Click **Submit**.

Save the changes to the flash memory with a device reset; after the device resets, the SAS menu appears and you can now begin configuring the SAS application.

19.2.1.2 Configuring Common SAS Parameters

The procedure below describes how to configure SAS settings that are common to all SAS modes. This includes various SAS parameters as well as configuring the Proxy Set for the SAS proxy (if required). The SAS Proxy Set ID defines the address of the UAs' external proxy.

To configure common SAS settings:

1. Open the SAS Configuration page (**Configuration** tab >**VoIP** menu >**SAS** > **Stand Alone Survivability**).

Define the port used for sending and receiving SAS messages. This can be any of the following port types:

UDP port - defined in the 'SAS Local SIP UDP Port' field

TCP port - defined in the 'SAS Local SIP TCP Port' field

TLS port - defined in the 'SAS Local SIP TLS Port' field

Note: This SAS port must be different than the device's local gateway port (i.e., that defined for the 'SIP UDP/TCP/TLS Local Port' parameter in the 'SIP General Parameters page page - **Configuration** tab >**VoIP** menu >**SIP Definitions**>**General Parameters**).

In the ‘SAS Default Gateway IP‘ field, define the IP address and port (in the format *x.x.x.x:port)* of the device. Note that the port of the device is defined by the parameter ‘SIP UDP Local Port’ (refer to the note in Step 2 above).

In the 'SAS Registration Time' field, define the value for the SIP Expires header, which is sent in the 200 OK response to an incoming REGISTER message when SAS is in emergency state.

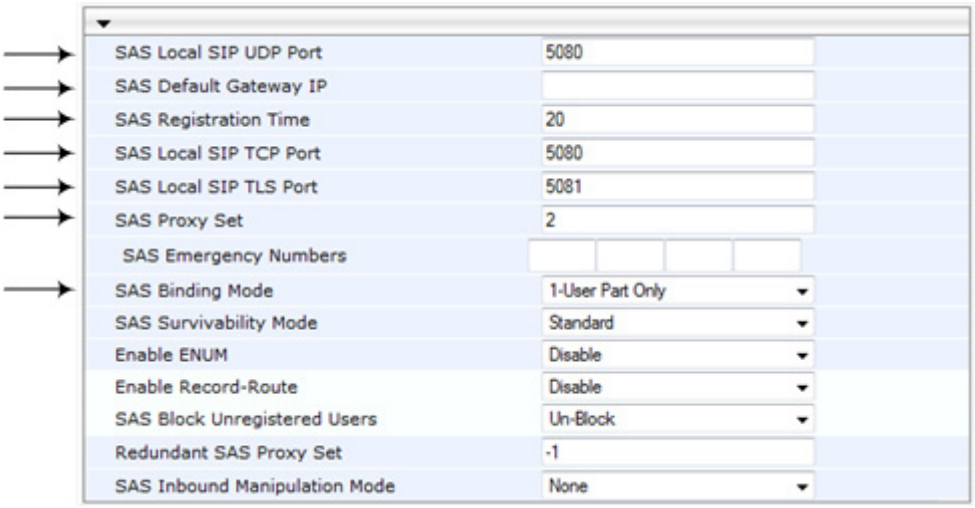
From the 'SAS Binding Mode' drop-down list, select the database binding mode:

0-URI: If the incoming AOR in the REGISTER request uses a ‘tel:’ URI or ‘user=phone’, the binding is done according to the Request-URI user part only. Otherwise, the binding is done according to the entire Request-URI (i.e., user and host parts - user@host).

1-User Part Only: Binding is done according to the user part only.

You must select **1-User Part Only** in cases where the UA sends REGISTER messages as SIP URI, but the INVITE messages sent to this UA include a Tel URI. For example, when the AOR of an incoming REGISTER is sip:3200@domain.com, SAS adds the entire SIP URI (e.g., sip:3200@domain.com) to its database (when the parameter is set to '0-URI'). However, if a subsequent Request-URI of an INVITE message for this UA arrives with sip:3200@10.1.2.3 user=phone, SAS searches its database for "3200", which it does not find. Alternatively, when this parameter is set to '1-User Part Only', then upon receiving a REGISTER message with sip:3200@domain.com, SAS adds only the user part (i.e., "3200") to its database. Therefore, if a Request-URI of an INVITE message for this UA arrives with sip:3200@10.1.2.3 user=phone, SAS can successfully locate the UA in its database.

Figure 19-9: Configuring Common Settings



In the 'SAS Proxy Set' field, enter the Proxy Set used for SAS. The SAS Proxy Set must be defined only for the following SAS modes:

Outbound mode: In SAS normal state, SAS forwards REGISTER and INVITE messages received from the UAs to the proxy servers defined in this Proxy Set.

Redundant mode and only if UAs don't support homing: SAS sends keep-alive messages to this proxy and if it detects that the proxy connection has resumed, it ignores the REGISTER messages received from the UAs, forcing them to send their messages directly to the proxy.

If you define a SAS Proxy Set ID, you must configure the Proxy Set as described in Step 8 below.

Click **Submit** to apply your settings.

If you defined a SAS Proxy Set ID in Step 6 above, then you must configure the SAS Proxy Set ID:

a. Open the Proxy Sets Table page (**Configuration** tab >**VoIP** menu >**Control Networks**>**Proxy Set Table**).

From the 'Proxy Set ID' drop-down list, select the required Proxy Set ID.

Notes:

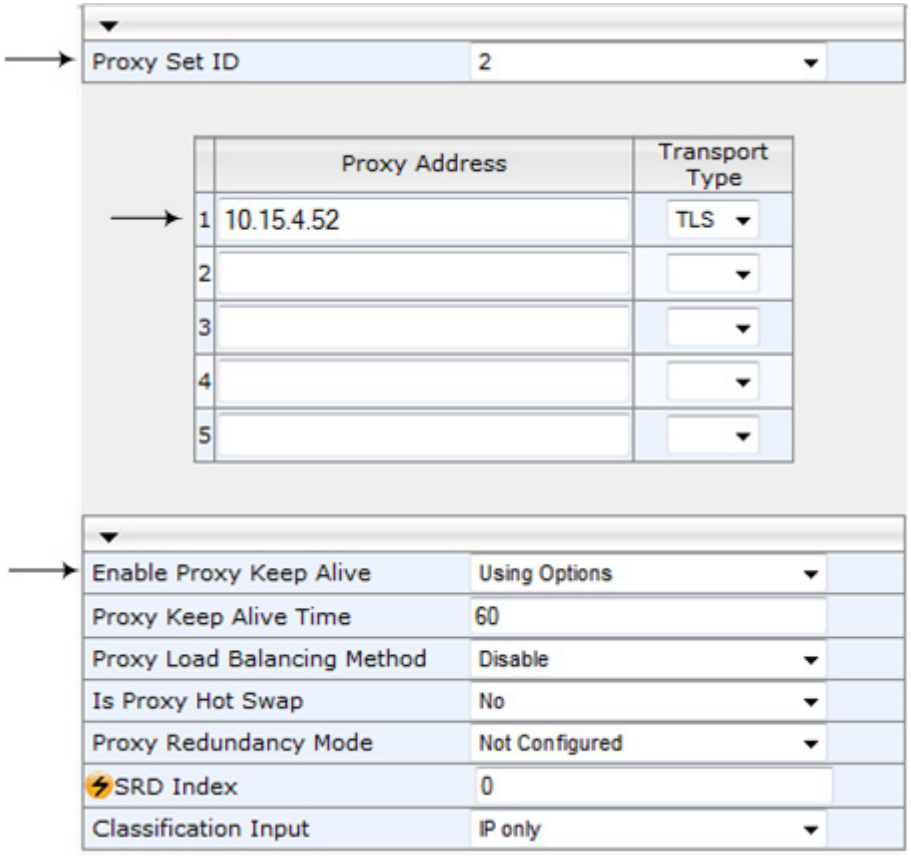
The selected Proxy Set ID number must be the same as that specified in the 'SAS Proxy Set' field in the 'SAS Configuration page (see Step 6).

Do not use Proxy Set ID 0.

In the 'Proxy Address' field, enter the IP address of the external proxy server.

From the 'Enable Proxy Keep Alive' drop-down list, select **Using Options**. This instructs the device to send SIP OPTIONS messages to the proxy for the keep-alive mechanism.

Figure 19-10: Defining UAs' Proxy Server



Click **Submit** to apply your settings.

19.2.2 Configuring SAS Outbound Mode

This section describes how to configure the SAS outbound mode. These settings are in addition to the ones described in 'Configuring Common SAS Parameters' on page 210.

Note: The VoIP CPEs (such as IP phones or residential gateways) need to be defined so that their proxy and registrar destination addresses and ports are the same as that configured for the device's SAS IP address and SAS local SIP port. In some cases, on the UAs, it is also required to define SAS as their outbound proxy, meaning that messages sent by the UAs include the host part of the external proxy, but are sent (on Layer 3/4) to the IP address / UDP port of SAS.

To configure SAS outbound mode:

1. Open the SAS Configuration page (**Configuration** tab >**VoIP** menu >**SAS** > **Stand Alone Survivability**).

From the 'SAS Survivability Mode' drop-down list, select **Standard**.

Click **Submit**.

19.2.3 Configuring SAS Redundant Mode

This section describes how to configure the SAS redundant mode. These settings are in addition to the ones described in 'Configuring Common SAS Parameters' on page 210.

Note: The VoIP CPEs (such as IP phones or residential gateways) need to be defined so that their primary proxy is the external proxy, and their redundant proxy destination addresses and port is the same as that configured for the device's SAS IP address and SAS SIP port.

To configure SAS redundant mode:

1. Open the SAS Configuration page (**Configuration** tab >**VoIP** menu >**SAS** >**Stand Alone Survivability**).

From the 'SAS Survivability Mode' drop-down list, select one of the following, depending on whether the UAs support homing (i.e., they always attempt to operate with the primary proxy, and if using the redundant proxy, they switch back to the primary proxy whenever it's available):

UAs support homing: Select **Always Emergency**. This is because SAS does not need to communicate with the primary proxy of the UAs; SAS serves only as the redundant proxy of the UAs. When the UAs detect that their primary proxy is available, they automatically resume communication with it instead of with SAS.

UAs do not support homing: Select **Ignore REGISTER**. SAS uses the keep-alive mechanism to detect availability of the primary proxy (defined by the SAS Proxy Set). If the connection with the primary proxy resumes, SAS ignores the messages received from the UAs, forcing them to send their messages directly to the primary proxy.

Click **Submit**.

19.2.4 Advanced SAS Configuration

This section describes the configuration of advanced SAS features that can be optionally implemented in your SAS deployment:

Manipulating incoming SAS Request-URI user part of REGISTER message (see 'Manipulating URI user part of Incoming REGISTER' on page 215)

Manipulating destination number of incoming SAS INVITE messages (see 'Manipulating Destination Number of Incoming INVITE' on page 217)

Defining SAS routing rules based on the IP2IP Routing table (see 'SAS Routing Based on IP2IP Routing Table' on page 221)

Blocking unregistered SAS UA's (see 'Blocking Calls from Unregistered SAS Users' on page 225)

Defining SAS emergency calls (see 'Configuring SAS Emergency Calls' on page 225)

Adding SIP Record-Route header to INVITE messages (see 'Adding SIP Record-Route Header to SIP INVITE' on page 226)

Replacing SIP Contact header (see 'Replacing Contact Header for SIP Messages' on page 226)

19.2.4.1 Manipulating URI user part of Incoming REGISTER

There are scenarios in which the UAs register to the proxy server with their full phone number (for example, "976653434"), but can receive two types of INVITE messages (calls):

INVITEs whose destination is the UAs' full number (when the call arrives from outside the enterprise)

INVITES whose destination is the last four digits of the UAs' phone number ("3434" in our example) when it is an internal call within the enterprise

Therefore, it is important that the device registers the UAs in the SAS registered database with their extension numbers (for example, "3434") in addition to their full numbers. To do this, you can define a manipulation rule to manipulate the SIP Request-URI user part of the AOR (in the To header) in incoming REGISTER requests. Once manipulated, it is saved in this manipulated format in the SAS registered users database in addition to the original (un-manipulated) AOR.

For example: Assume the following incoming REGISTER message is received and that you want to register in the SAS database the UA's full number as well as the last four digits from the right of the SIP URI user part:

```
REGISTER sip:10.33.38.2 SIP/2.0
Via: SIP/2.0/UDP 10.33.4.226:5050;branch=z9hG4bKac10827
Max-Forwards: 70
From: <sip: 976653434@10.33.4.226>;tag=1c30219
To: <sip: 976653434@10.33.4.226>
Call-ID: 16844@10.33.4.226
CSeq: 1 REGISTER
Contact: <sip: 976653434@10.10.10.10:5050>;expires=180
Allow:
REGISTER,OPTIONS,INVITE,ACK,CANCEL,BYE,NOTIFY,PRACK,REFER,INFO,SUBSCRIBE,UPD
ATE
Expires: 180
User-Agent: Alcatel-Lucent-Sip-Gateway-/v.
Content-Length: 0
```

After manipulation, SAS registers the user in its database as follows:

AOR: 976653434@10.33.4.226

Associated AOR: 3434@10.33.4.226 (after manipulation, in which only the four digits from the right of the URI user part are retained)

Contact: 976653434@10.10.10.10

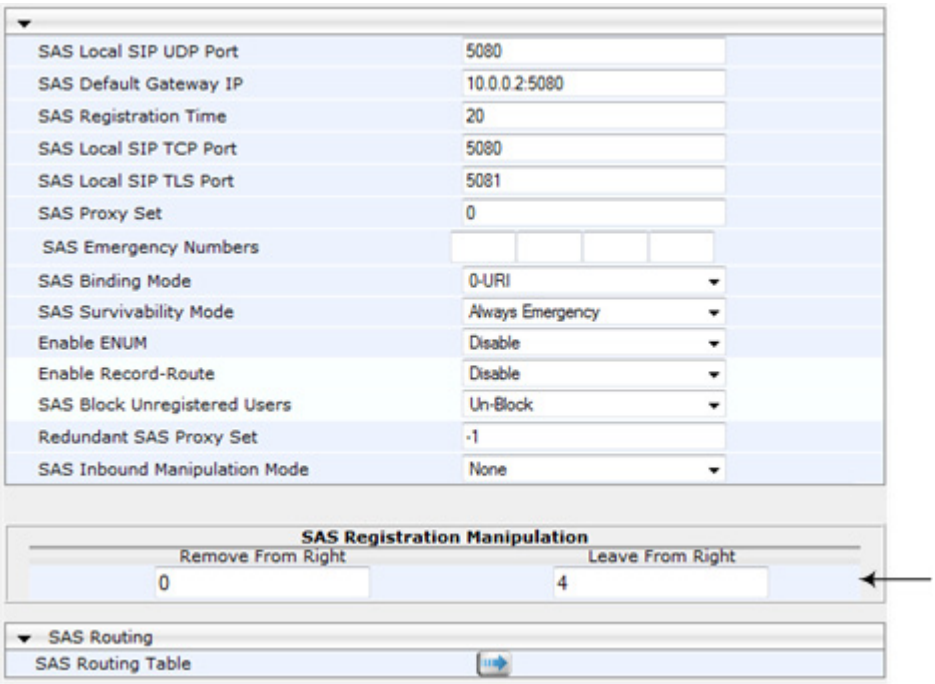
The procedure below describes how to configure the manipulation example scenario above (relevant *ini* parameter is SASRegistrationManipulation):

To manipulate incoming Request-URI user part of REGISTER message:

1. Open the SAS Configuration page (**Configuration** tab >**VoIP** menu >**SAS** >**Stand Alone Survivability**).

In the SAS Registration Manipulation table, in the 'Leave From Right' field, enter the number of digits (e.g., "4") to leave from the right side of the user part. (The Leave From Right' field defines the number of digits to retain from the right side of the user part; all other digits in the user part are removed.)

Figure 19-11: Manipulating User Part in Incoming REGISTER



Click **Submit**.

19.2.4.2 Manipulating Destination Number of Incoming INVITE

You can define a manipulation rule to manipulate the destination number in the Request-URI of incoming INVITE messages when SAS is in emergency state. This is required, for example, if the call is destined to a registered user but the destination number in the received INVITE is not the number assigned to the registered user in the SAS registration database. To overcome this and successfully route the call, you can define manipulation rules to change the INVITE's destination number so that it matches that of the registered user in the database. This is done using the IP to IP Inbound Manipulation table.

For example, in SAS emergency state, assume an incoming INVITE has a destination number "7001234" which is destined to a user registered in the SAS database as "552155551234". In this scenario, the received destination number needs to be manipulated to the number "552155551234". The outgoing INVITE sent by the device then also contains this number in the Request-URI user part.

In normal state, the numbers are not manipulated. In this state, SAS searches the number 552155551234 in its database and if found, it sends the INVITE containing this number to the UA.

To manipulate destination number in SAS emergency state:

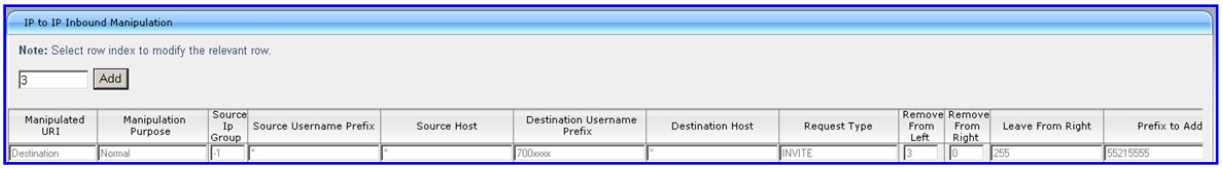
2. Open the SAS Configuration page (**Configuration**tab >**VoIP** menu >**SAS** >**Stand Alone Survivability**).

From the 'SAS Inbound Manipulation Mode' (*SASInboundManipulationMode*) drop-down list, select **Emergency Only**.

Click **Submit**; the **SAS Inbound Manipulation Mode Table** button appears on the page.

Click this button to open the IP to IP Inbound Manipulation page.

Figure 19-12: Manipulating INVITE Destination Number



The figure above displays a manipulation rule for the example scenario described above whereby the destination number "7001234" is changed to "552155551234":

'Manipulated URI' field: **Destination**

'Destination Username Prefix' field: "700xxxx"

'Request Type' field: **INVITE**

'Remove From Left' field: "3"

'Prefix to Add' field: "55215555"

Add your SAS manipulation rule as desired. See the table below for descriptions of the parameters.

Click **Apply** to save your changes.

Notes: The following fields of the IP to IP Inbound Manipulation table are not applicable to SAS and must be left at their default values:

Additional Manipulation - default is "0"

Manipulation Purpose - default is Normal

Source IP Group - default is "-1"

Table 19-1: SAS IP to IP Inbound Manipulation Parameters

Parameter	Description
Matching Characteristics	
Is Additional Manipulation **[IsAdditionalManipulation]**	Determines whether additional SIP URI user part manipulation is done for the table entry rule listed directly above it. **[0]** 0 = Regular manipulation rule (not done in addition to the rule above it). **[1]** 1 = If the above row entry rule matched the call, consider this row entry as a match as well and perform the manipulation specified by this rule. **Note:** Additional manipulation can only be performed on a different SIP URI (either source or destination) to the rule configured in the row above (defined by the parameter ManipulatedURI).
Manipulation Purpose **[ManipulationPurpose]**	Defines the purpose of the manipulation: **[0]** Normal = Inbound manipulations affect the routing input and source and/or destination number (default). **[1]** Routing input only = Inbound manipulations affect the routing input only, retaining the original source and destination number. **[2]** Shared Line = Used for the Shared-Line Appearance feature. This manipulation is for registration requests to change the destination number of the secondary extension numbers to the primary extension.
Source IP Group **[SrcIpGroup]**	Defines the IP Group from where the incoming INVITE is received. For any Source IP Group, enter the value -1.
Source Username Prefix **[SrcUsernamePrefix]**	Defines the prefix of the source SIP URI user name (usually in the From header). For any prefix, enter the asterisk "*" symbol (default). **Note:** The prefix can be a single digit or a range of digits. For available notations, see "Dialing Plan Notation for Routing and Manipulation" on page 190.
Source Host **[SrcHost]**	Defines the source SIP URI host name - full name (usually in the From header). For any host name, enter the asterisk "*" symbol (default).
Destination Username Prefix **[DestUsernamePrefix]**	Defines the prefix of the destination SIP URI user name (usually in the Request-URI). For any prefix, enter the asterisk "*" symbol (default). **Note:** The prefix can be a single digit or a range of digits. For available notations, see "Dialing Plan Notation for Routing and Manipulation" on page 190.
Destination Host **[DestHost]**	Defines the destination SIP URI host name - full name (usually in the Request URI). For any host name, enter the asterisk "*" symbol (default).
Request Type **[RequestType]**	Defines the SIP request type to which the manipulation rule is applied. **[0]** All = all SIP messages (default) **[1]** INVITE = all SIP messages except REGISTER and SUBSCRIBE **[2]** REGISTER = only SIP REGISTER messages **[3]** SUBSCRIBE = only SIP SUBSCRIBE messages **[4]** INVITE and REGISTER = all SIP messages except SUBSCRIBE **[5]** INVITE and SUBSCRIBE = all SIP messages except REGISTER
Manipulated URI **[ManipulatedURI]**	Determines whether the source or destination SIP URI user part is manipulated. **[0]** Source = Manipulation is done on the source SIP URI user part. (default) **[1]** Destination = Manipulation is done on the destination SIP URI user part.
Operation Manipulation Rule (when match occurs in characteristics)	

Parameter	Description
Remove From Left **[RemoveFromLeft]**	Defines the number of digits to remove from the left of the user name prefix. For example, if you enter 3 and the user name is "john", the new user name is "n".
Remove From Right **[RemoveFromRight]**	Defines the number of digits to remove from the right of the user name prefix. For example, if you enter 3 and the user name is "john", the new user name is "j".
Leave From Right **[LeaveFromRight]**	Defines the number of characters that you want retained from the right of the user name.
Prefix to Add **[Prefix2Add]**	Defines the number or string that you want added to the front of the user name. For example, if you enter 'user' and the user name is "john", the new user name is "userjohn".
Suffix to Add **[Suffix2Add]**	Defines the number or string that you want added to the end of the user name. For example, if you enter '01' and the user name is "john", the new user name is "john01".

19.2.4.3 SAS Routing Based on IP2IP Routing Table

SAS routing based on rules configured in the SAS Routing table is applicable for SAS in the following states:

SAS in normal state, if the SASSurvivabilityMode parameter is set to 4

SAS in emergency state, if the SASSurvivabilityMode parameter is not set to 4

The SAS routing rule destination can be an IP Group, IP address, Request-URI, or ENUM query.

The IP2IP Routing Table page allows you to configure up to 120 SAS routing rules (for Normal and Emergency modes). The device routes the SAS call (received SIP INVITE message) once a rule in this table is matched. If the characteristics of an incoming call do not match the first rule, the call characteristics is then compared to the settings of the second rule, and so on until a matching rule is located. If no rule is matched, the call is rejected.

When SAS receives a SIP INVITE request from a proxy server, the following routing logic is performed:

a. Sends the request according to rules configured in the IP2IP Routing table.

If no matching routing rule exists, the device sends the request according to its SAS registration database.

If no routing rule is located in the database, the device sends the request according to the Request-URI header.

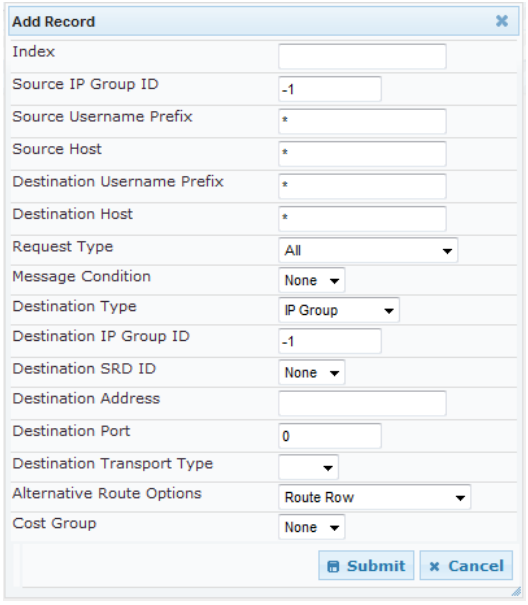
Note: The IP2IP Routing table can also be configured using the *ini* file table parameter IP2IPRouting (see 'Configuration Parameters Reference' on page **Erreur ! Signet non défini.**).

To configure the IP2IP Routing table for SAS:

3. In the SAS Configuration page, click the **SAS Routing Table** button; the IP2IP Routing Table page appears.

Click **Add**.

Figure 19-13: Add Record Dialog Box for Adding SAS Routing Rule



Configure the rule according to the table below.

Click **Submit** to apply your changes.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

Note: The following parameters are not applicable to SAS and should be ignored: Source IP Group ID, Destination IP Group ID, Destination SRD ID, and Alternative Route Options.

Table 19-2: SAS IP2IP Routing Table Parameters

Parameter	Description
Matching Characteristics	
Source IP Group ID **[IP2IPRouting_SrcIPGroupID]**	Selects the IP Group from where the IP-to-IP call originated. Typically, the IP Group of an incoming SIP dialog is determined (or classified) using the 'Classification' table (see "Configuring Classification Table" on page 178). If not used (i.e., any IP Group), simply leave the field empty. The default is -1.
Source Username Prefix **[IP2IPRouting_SrcUsernamePrefix]**	Defines the prefix of the user part of the incoming SIP dialog's source URI (usually the From URI). The default is "*". **Note:** The prefix can be a single digit or a range of digits. For available notations, see "Dialing Plan Notation for Routing and Manipulation" on page 190.
Source Host **[IP2IPRouting_SrcHost]**	Defines the host part of the incoming SIP dialog's source URI (usually the From URI). If this rule is not required, leave the field empty. To denote any host name, use the asterisk (*) symbol. The default is "*".
Destination Username Prefix **[IP2IPRouting_DestUsernamePrefix]**	Defines the prefix of the incoming SIP dialog's destination URI (usually the Request URI) user part. If this rule is not required, leave the field empty. To denote any prefix, use the asterisk (*) symbol. The default is "*". **Note:** The prefix can be a single digit or a range of digits. For available notations, see "Dialing Plan Notation for Routing and Manipulation" on page 190.
Destination Host **[IP2IPRouting_DestHost]**	Defines the host part of the incoming SIP dialog's destination URI (usually the Request-URI). If this rule is not required, leave the field empty. The asterisk (*) symbol can be used to denote any destination host. The default is "*".
Request Type **[IP2IPRouting_RequestType]**	Defines the SIP dialog request type of the incoming SIP dialog. **[0]** All (default) **[1]** INVITE **[2]** REGISTER **[3]** SUBSCRIBE **[4]** INVITE and REGISTER **[5]** INVITE and SUBSCRIBE **[6]** OPTIONS
Message Condition **[IP2IPRouting_MessageCondition]**	Selects a Message Condition rule. To configure Message Condition rules, see "Configuring Condition Rules" on page 182.

Parameter	Description
Operation Routing Rule (when match occurs in characteristics)	
Destination Type **[IP2IPRouting_DestType]**	Determines the destination type to which the outgoing SIP dialog is sent. **[0]** IP Group (default) = The SIP dialog is sent to the IP Group's Proxy Set (SERVER-type IP Group) or registered contact from the database (if USER-type IP Group). **[1]** Dest Address = The SIP dialog is sent to the address configured in the following fields: 'Destination SRD ID', 'Destination Address', 'Destination Port', and 'Destination Transport Type'. **[2]** Request URI = The SIP dialog is sent to the address indicated in the incoming Request-URI. If the fields 'Destination Port' and 'Destination Transport Type' are configured, the incoming Request-URI parameters are overridden and these fields take precedence. **[3]** ENUM = An ENUM query is sent to include the destination address. If the fields 'Destination Port' and 'Destination Transport Type' are configured, the incoming Request-URI parameters are overridden and these fields take precedence. **[4]** Hunt Group = Used for call center survivability. For more information, see "Call Survivability for Call Centers" on page 167.
Destination IP Group ID **[IP2IPRouting_DestIPGroupID]**	Defines the IP Group ID to where you want to route the call. The SIP dialog messages are sent to the IP address defined for the Proxy Set associated with this IP Group. If you select an IP Group, it is unnecessary to configure a destination IP address (in the 'Destination Address' field). However, if both parameters are configured, then the IP Group takes precedence. If the destination IP Group is of USER type, the device searches for a match between the Request-URI (of the received SIP dialog) to an AOR registration record in the device's database. The SIP dialog is then sent to the IP address of the registered contact. The default is -1. **Notes:** This parameter is only relevant if the parameter 'Destination Type' is set to 'IP Group'. However, regardless of the settings of the parameter 'Destination Type', the IP Group is still used - only for determining the IP Profile or outgoing SRD. If neither IP Group nor SRD are defined in this table, the destination SRD is determined according to the source SRD associated with the Source IP Group (configured in the IP Group table, see "Configuring IP Groups" on page 114). If this table does not define an IP Group but only an SRD, then the first IP Group associated with this SRD (in the IP Group table) is used. If the selected destination IP Group ID is type SERVER, the request is routed according to the IP Group addresses. If the selected destination IP Group ID is type USER, the request is routed according to the IP Group specific database (i.e., only to registered users of the selected database). If the selected destination IP Group ID is ANY USER (**[-2**]), the request is routed according to the general database (i.e., any matching registered user).
Destination SRD ID **[IP2IPRouting_DestSRDID]**	Defines the SRD ID. The default is None. **Note:** The destination IP Group must belong to the destination SRD, if both are configured in this table.

Parameter	Description
Destination Address **[IP2IPRouting_DestAddress]**	Defines the destination IP address (or domain name, e.g., domain.com) to where the call is sent. **Notes:** This parameter is applicable only if the parameter 'Destination Type' is set to 'Dest Address' **[1].** When using domain names, enter a DNS server IP address or alternatively, define these names in the 'Internal DNS Table' (see "Configuring the Internal SRV Table" on page 92).
Destination Port **[IP2IPRouting_DestPort]**	Defines the destination port to where the call is sent.
Destination Transport Type **[IP2IPRouting_DestTransportType]**	Defines the transport layer type for sending the call: **[-1]** Not Configured (default) **[0]** UDP **[1]** TCP **[2]** TLS **Note:** When this parameter is set to -1, the transport type is determined by the parameter SIPTransportType.
Alternative Route Options **[IP2IPRouting_AltRouteOptions]**	Determines whether this routing rule is the main routing rule or an alternative routing rule (to the rule defined directly above it in the table). **[0]** Route Row (default) = Main routing rule - the device first attempts to route the call to this route if the incoming SIP dialog's input characteristics matches this rule. **[1]** Alt Route Ignore Inputs = If the call cannot be routed to the main route (Route Row), the call is routed to this alternative route regardless of the incoming SIP dialog's input characteristics. **[2]** Alt Route Consider Inputs = If the call cannot be routed to the main route (Route Row), the call is routed to this alternative route only if the incoming SIP dialog matches this routing rule's input characteristics. **Notes:** The alternative routing entry (**[1]** or **[2]) must be defined in the next consecutive table entry index** to the Route Row entry (i.e., directly below it). For example, if Index 4 is configured as a Route Row, Index 5 must be configured as the alternative route. For IP-to-IP alternative routing, configure SBC alternative routing reasons upon receipt of 4xx, 5xx, and 6xx SIP responses (see "Configuring Alternative Routing Reasons" on page 189). However, if no response, ICMP, or a SIP 408 response is received, the device attempts to use the alternative route even if no entries are configured in the 'SBC Alternative Routing Reasons' table. Multiple alternative route entries can be configured (e.g., Index 1 is the main route - Route Row - and indices 2 through 4 are configured as alternative routes).
Cost Group **[IP2IPRouting_CostGroup]**	Reserved for future use.

19.2.4.4 Blocking Calls from Unregistered SAS Users

To prevent malicious calls (for example, Service Theft), it is recommended to configure the feature for blocking SIP INVITE messages received from SAS users that are not registered in the SAS database. This applies to SAS in normal and emergency states.

To block calls from unregistered SAS users:

1. Open the SAS Configuration page (**Configuration** tab >**VoIP** menu >**SAS Stand Alone Survivability**).

From the 'SAS Block Unregistered Users' drop-down list, select **Block**.

Click **Submit** to apply your changes.

19.2.4.5 Configuring SAS Emergency Calls

You can configure SAS to route emergency calls (such as 911 in North America) directly to the PSTN (through another gateway). Therefore, even during a communication failure with the external proxy, enterprise UAs can still make emergency calls.

You can define up to four emergency numbers, where each number can include up to four digits. When SAS receives a SIP INVITE (from a UA) that includes one of the user-defined emergency numbers in the SIP user part, it forwards the INVITE directly to the default gateway (see 'SAS Routing in Emergency State' on page 208). The default gateway is defined in the 'SAS Default Gateway IP' field (this can be the device itself). The gateway then sends the call directly to the PSTN.

This feature is applicable to SAS in normal and emergency states.

To configure SAS emergency numbers:

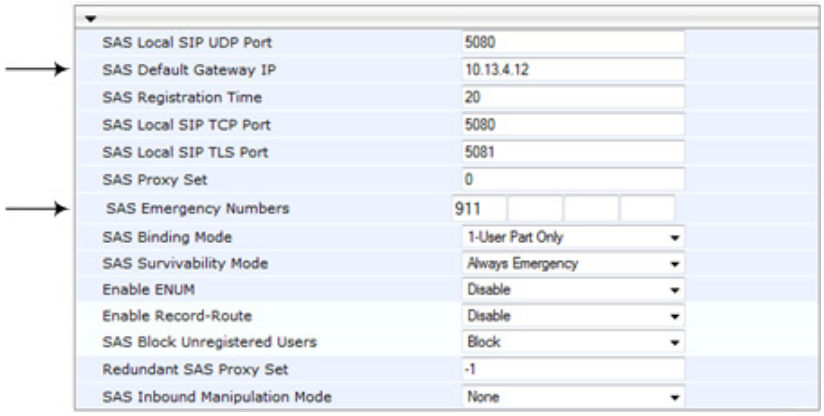
1. Open the SAS Configuration page (**Configuration** tab >**VoIP** menu >**SAS** >**Stand Alone Survivability**).

In the 'SAS Default Gateway IP' field, define the IP address and port (in the format *x.x.x.x:port*) of the gateway.

Note: The port of the device is defined in the 'SIP UDP/TCP/TLS Local Port' field in the SIP General Parameters page (**Configuration** tab >**VoIP** menu >**SIP Definitions** >**General Parameters**).

In the 'SAS Emergency Numbers' field, enter an emergency number in each field box.

Figure 19-14: Configuring SAS Emergency Numbers



Click **Submit** to apply your changes.

19.2.4.6 Adding SIP Record-Route Header to SIP INVITE

You can configure SAS to add the SIP Record-Route header to SIP requests (e.g. INVITE) received from enterprise UAs. SAS then sends the request with this header to the proxy. The Record-Route header includes the IP address of the SAS application. This ensures that future requests in the SIP dialog session from the proxy to the UAs are routed through the SAS application. If not configured, future request within the dialog from the proxy are sent directly to the UAs (and do not traverse SAS). When this feature is enabled, the SIP Record-Route header includes the URI "lr" parameter, indicating loose routing, as shown in the following example:

```
Record-Route: <sip:server10.biloxi.com;lr>
```

Notes:

This feature is applicable only to SAS outbound mode.

This feature can also be enabled using the SASEnableRecordRoute *ini* file parameter.

To enable the Record-Route header:

1. Open the SAS Configuration page (**Configuration** tab >**VoIP** menu >**SAS** >**Stand Alone Survivability**).

From the 'Enable Record-Route' drop-down list, select **Enable**.

Click **Submit** to apply your changes.

19.2.4.7 Replacing Contact Header for SIP Messages

You can configure SAS to change the SIP Contact header so that it points to the SAS host. Therefore, this ensures that in the message, the top-most SIP Via header and the Contact header point to the same host.

Notes:

This feature is applicable only to SAS outbound mode.

The device may become overloaded if this feature is enabled, as all incoming SIP dialog requests traverse the SAS application.

Currently, this feature can only be configured using the SASEnableContactReplace *ini* file parameter.

[0] (default): Disable - when relaying requests, SAS adds a new Via header (with the IP address of the SAS application) as the top-most Via header and retains the original Contact header. Thus, the top-most Via header and the Contact header point to different hosts.

[1]: Enable - SAS changes the Contact header so that it points to the SAS host and therefore, the top-most Via header and the Contact header point to the same host.

19.3 Viewing Registered SAS Users

You can view all the users that are registered in the SAS registration database. This is displayed in the 'SAS/SBC Registered Users page, as described in 'Viewing SAS/SBC Registered Users' on page 270.

Note: Despite the maximum number of SAS users, you can increase this capacity by implementing the SAS Cascading feature, as described in 'SAS Cascading' on page 227.

19.4 SAS Cascading

The SAS Cascading feature allows you to increase the number of SAS users above the maximum supported by the SAS gateway. This is achieved by deploying multiple SAS gateways in the network. For example, if the SAS gateway supports up to 600 users, but your enterprise has 1,500 users, you can deploy three SAS gateways to accommodate all users: the first SAS gateway can service 600 registered users, the second SAS gateway the next 600 registered users, and the third SAS gateway the rest (i.e., 300 registered users).

In SAS Cascading, the SAS gateway first attempts to locate the called user in its SAS registration database. Only if the user is not located, does the SAS gateway send it on to the next SAS gateway according to the SAS Cascading configuration.

There are two methods for configuring SAS Cascading. This depends on whether the users can be identified according to their phone extension numbers:

SAS Routing Table: If users can be identified with unique phone extension numbers, then the SAS Routing table is used to configure SAS Cascading. This SAS Cascading method routes calls directly to the SAS Gateway (defined by IP address) to which the called SAS user is registered.

The following is an example of a SAS Cascading deployment of users with unique phone extension numbers:

users registered to the first SAS gateway start with extension number “40”

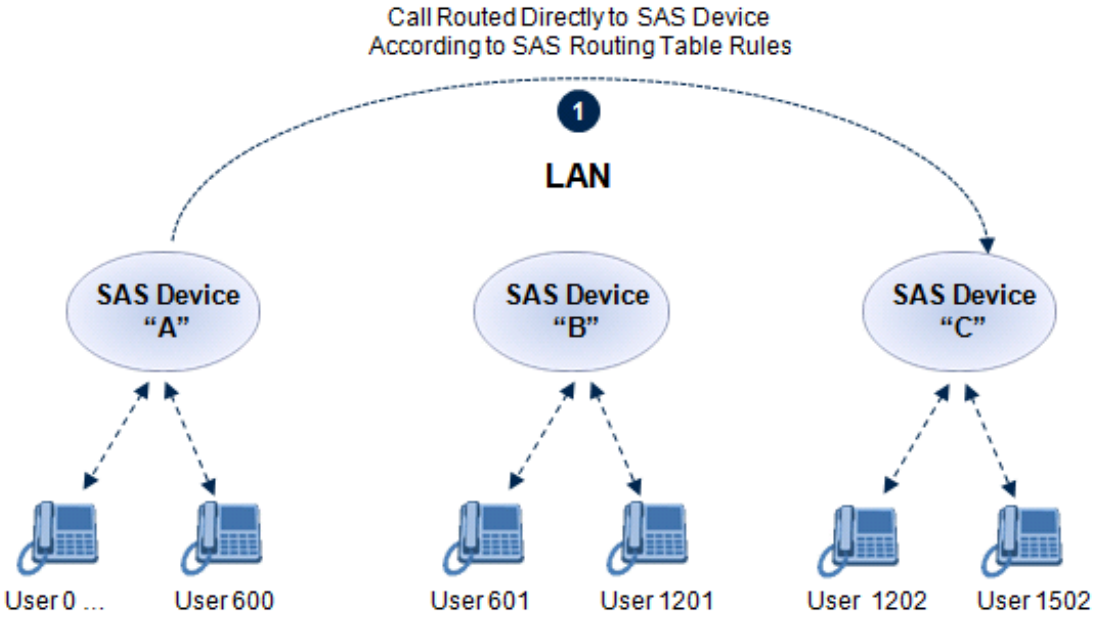
users registered to the second SAS gateway start with extension number “20”

users registered to the third SAS gateway start with extension number “30”

The SAS Routing table rules for SAS Cascading are created using the destination (called) extension number prefix (e.g., “30”) and the destination IP address of the SAS gateway to which the called user is registered. Such SAS routing rules must be configured at each SAS gateway to allow routing between the SAS users. The routing logic for SAS Cascading is similar to SAS routing in Emergency state (see the flowchart in 'SAS Routing in Emergency State' on page 208). For a description on the SAS Routing table, see 'SAS Routing Based on SAS Routing Table' on page 221.

The figure below illustrates an example of a SAS Cascading call flow configured using the SAS Routing table. In this example, a call is routed from SAS Gateway (A) user to a user on SAS Gateway (B).

Figure 19-15: SAS Cascading Using SAS Routing Table - Example

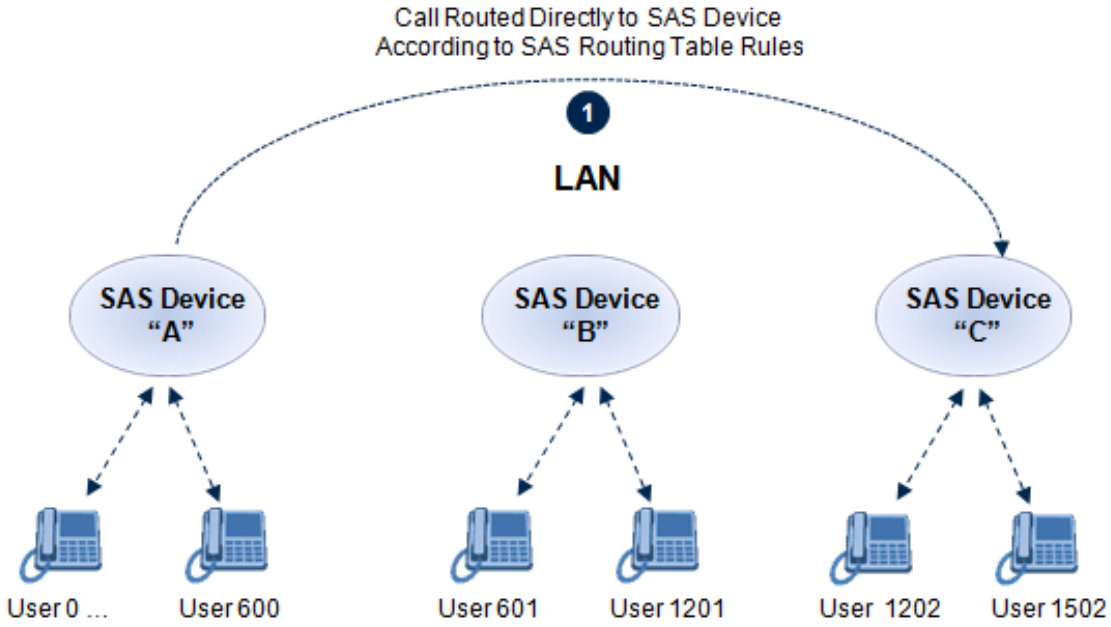


SAS Redundancy mode: If users cannot be distinguished (i.e., associated to a specific SAS gateway), then the SAS Redundancy feature is used to configure SAS Cascading. This mode routes the call in a loop fashion, from one SAS gateway to the next, until the user is located. Each SAS gateway serves as the redundant SAS gateway ("redundant SAS proxy server") for the previous SAS gateway (in a one-way direction). For example, if a user calls a user that is not registered on the same SAS gateway, the call is routed to the second SAS gateway, and if not located, it is sent to the third SAS gateway. If the called user is not located on the third (or last) SAS gateway, it is then routed back to the initial SAS gateway, which then routes the call to the default gateway (i.e., to the PSTN).

Each SAS gateway adds its IP address to the SIP via header in the INVITE message before sending it to the next ("redundant") SAS gateway. If the SAS gateway receives an INVITE and its IP address appears in the SIP via header, it sends it to the default gateway (and not to the next SAS gateway), as defined by the SASDefaultGatewayIP parameter. Therefore, this mode of operation prevents looping between SAS gateways when a user is not located on any of the SAS gateways.

The figure below illustrates an example of a SAS Cascading call flow when configured using the SAS Redundancy feature. In this example, a call is initiated from a SAS Gateway (A) user to a user that is not located on any SAS gateway. The call is subsequently routed to the PSTN.

Figure 19-16: SAS Cascading Using SAS Redundancy Mode - Example



Part V

High Availability

This part describes the configuration of the High Availability system.

20 Overview

The device's High Availability (HA) feature provides full redundancy between two OpenTouch™ Session Border Controller devices.

In HA mode, one of the LAN interfaces (Ethernet Group) on each device is used for the Ethernet connectivity between the two devices and is referred to as the *Maintenance* interface. The devices must be connected through their Maintenance interface to each other on the same broadcast domain. This connection can be one of the following:

A direct connection (i.e., port to port). However, in this setup, the physical port group used for this connection can only be used for the Maintenance interface.

Indirect connection through a switch. In this setup, the physical port group used for this connection can also be used for other interfaces (i.e., OAMP, Media, and/or Control) in addition to the Maintenance interface.

Each device has its own Maintenance interface with a unique address and each device is familiar with the Maintenance address of the remote device.

Under normal operation, one of the devices is in “Active” state while the second device is in "Redundant" state. In the Active device, all logical interfaces are active (i.e., Media, Control, OAMP, Maintenance, etc…). In the Redundant device, only the Maintenance interface is active (used for connectivity with the Active device). Therefore, management of the device is done only through the Active device.

Upon a major functional failure in the Active device, the Redundant device becomes active and activates all its logical interfaces exactly as was configured in the Active device.

Note: It is recommended to avoid using Spanning Tree Protocol (STP) on the interface used for Maintenance. The Ethernet connectivity of the Maintenance interface between the two devices should be constantly reliable without any disturbances.

20.1 Revertive Mode

The HA system can be set to the Revertive mode, which allows specifying one of the devices as the favorite or prioritized device between the two devices. When operating in Revertive mode, each device is configured with a priority level between 1 and 10 (where 1 is the lowest). Whenever the device that is set with a higher priority recovers from a failure, it first becomes the redundant device and then it issues an automatic switchover to become the active device once again (otherwise, after recovery, it becomes the redundant device and remains as redundant).

If you increase the priority of the redundant device to a level that is higher than the active device and then reset the redundant device, then a switchover occurs to the redundant device which becomes the active device.

If both devices are configured with the same priority level, then Revertive mode is irrelevant.

When Revertive mode is disabled, a switchover is done only upon failure of the active device.

20.2 Initialization Process

When only one device is running, it is in HA stand-alone state, i.e., the Active device without connectivity to the Redundant device. When the second device is loaded it recognizes the Active device through the Maintenance network and acquires the HA Redundant state. It then begins synchronizing for HA with the Active device.

Synchronization between the Active and Redundant devices may take several minutes in which the Active device provides the Redundant device with all its current configuration settings (including loaded files). In addition, the Active device also provides it with the software (.cmp file) if the Redundant device is loaded with a different software version. Once loaded to the Redundant device, the Redundant device reboots to apply the new configuration.

20.3 HA Status in the Home Page

When the device operates in HA mode, the Home page displays "Operational" in the 'High Availability' field.

20.4 Device Failure Detection

Constant keep-alive messages are sent between both devices to verify connectivity. Upon detection of a device failure, the following occurs:

Active device failure: The Redundant device issues a switch-over operation. As part of this switch-over operation, the failed device resets and the previously Redundant device becomes the Active device in stand-alone mode until a Redundant device is detected. The previously Active device reboots. If the failure in the Active device is repaired after reset, it is initialized as the Redundant device and the system returns to HA mode.

Redundant device failure: The Active device moves itself into stand-alone mode until the Redundant device is returned to functional operation. If the failure in the Redundant device is repaired after reset, it's initialized as the Redundant device once again and the system returns to HA mode.

21 HA Configuration

This section describes the configuration of the HA system.

21.1 Initial HA Configuration

By default, HA is disabled. When a device is loaded with valid HA configuration, if it is the first device to be loaded it becomes the active device. The second device that is loaded with HA configuration becomes the redundant (standby) device. If Revertive mode is enabled, then when a redundant device with higher priority is loaded, it issues a switchover to gain control of the system.

21.1.1 Stage 1: Installation of Both Devices

Follow the installation instructions as described in Chapter 2 of the *Installation Manual*. You must assign each device a temporary address, using the CLI,which is used for first connection to the Web management of the devices.

In the following sections, the first device is configured with IP address 10.0.0.1 and the second device is configured with IP address 10.0.0.2.

21.1.2 Stage 2: Connect the Devices to Same Network Topology

This stage describes physical connection between the devices.

To connect the devices in the same network topology:

2. Connect the physical ports of each device in the same way (first port of both devices is connected to the same broadcast domain and the second port of both devices is connected to the same broadcast domain).

Ensure that the physical ports used for the Maintenance interface on both devices are connected to each other (according to your network topology - directly or indirectly).

21.1.3 Stage 3: Configure the First Device

The first stage is to configure the first device for HA, as described below.

To configure the first device for HA:

3. Connect to the Web interface of the first device, using its OAMP network address (for example, 10.0.0.1).

Perform regular device configuration as desired as if it were a stand-alone unit.

Add the HA Maintenance interface:

a. Open the Multiple Interface table (**Configuration** tab >**VoIP** menu >**Network** submenu >**IP Settings**). For more information on configuring IP interfaces, see Configuring IP Interface Settings.

Add an interface for the 'Application Type' **MAINTENANCE**.

Configure the HA parameters:

a. Open the HA Settings page (**Configuration** tab >**System** menu >**HA Settings**).

In the 'HA Remote Address' field, enter the Maintenance IP address ofthe second device. This must be on the same subnet as the configured address of the Maintenance interface (configured in Step 3).

(Optional) Enable the HA Revertive mode by selecting **Enable** from the 'HA Revertive' drop-down list, and then in the 'HA Priority' field, enter the HA priority level of this device. The 'Redundant HA Priority' field is only later used in the Web interface of the redundant device, or in the Web interface of this active device once HA is already up and running.

Reset the device (see 'Resetting the Device' on page 242) and ensure that it is operating normally as a standalone unit.

21.1.4 Stage 4: Configure the Second Device

This stage configures the second device for HA, as described below.

Note: After you configure the second device for HA, its regular device configuration (i.e., not related to HA) is received later from the first (Active) device during HA synchronization.

To configure the second device for HA:

4. Connect to the Web interface of the second device, using its OAMP network address (for example, 10.0.0.2).

Add the HA Maintenance interface:

a. Open the Multiple Interface table (**Configuration** tab >**VoIP** menu >**Network** submenu >**IP Settings**). For more information on configuring IP interfaces, see Configuring IP Interface Settings.

Add an interface for the 'Application Type' **MAINTENANCE**.

Note: The Maintenance interface must be configured with the same settings (e.g., same VLAN ID and Ethernet port group) as the first device, except for the IP address which must be the same as the HA Remote address configured in the first device.

Configure the HA parameters:

a. Open the HA Settings page (**Configuration** tab >**System** menu >**HA Settings**).

In the 'HA Remote Address' field, enter the Maintenance address of the first device. This must be on the same subnet as the configured address of the Maintenance interface (configured in Step 3).

(Optional) Enable the HA Revertive mode by selecting **Enable** from the 'HA Revertive' drop-down list, and then in the 'Redundant HA Priority' field, enter the HA priority level of the second device.

Reset the second device (see 'Resetting the Device' on page 242) to apply changes.

21.1.5 Stage 5: Booting Up Second Device

At this final stage, the second device boots up and synchronizes with the first device.After the second device boots-up, it starts synchronizing with the first device and updates its configuration according to first device.

During synchronization, the Web interface Home page of the first device displays the HA status as "Synchronizing".

When synchronization completes successfully, the second device is reloaded to apply the received configuration.

When both devices are up and running, the Web interface Home page displays the HA status as "Operational". At this stage, the first device is active and the second device is redundant.

21.2 Configuration while HA State is Operational

When the device is up and running in HA state, configuration is as follows:

All configurations (including HA) is done through the Active device **only**.

Non-HA configuration done on the Active device is automatically updated to the Redundant device.

HA configuration done on the Active device is updated to the Redundant device as follows:

- Configuring a new Maintenance interface address for the Active device is automatically set as the new 'HA Remote Address' value in the Redundant device. Configuring a new 'HA Remote Address' value on the Active device automatically sets it as the new Maintenance interface address in the Redundant device (this requires a device reset). All other Maintenance interface parameters (e.g., Gateway address and VLAN ID) are updated also to the Maintenance interface of the Redundant device.
- 'HA Revertive' mode is updated to the Redundant device (this requires a device reset).
- The 'HA Priority' parameter is set for the Active device and the 'Redundant HA Priority' parameter is set for the Redundant device (requires a device reset).

Tip: If the HA system is already in Revertive mode and you want to change the prioritized device, to ensure that system service is maintained and traffic is not disrupted, it is recommended to set the higher priority to the Redundant device and then reset it. After it synchronizes with the Active device, it issues a switch over and becomes the new Active device (the original Active device resets and becomes the new Redundant device).

Changing the subnet network of the HA Maintenance interface must be done separately on each device, as described below.

To change the subnet of the Maintenance interface:

5. On the Redundant device (A):

a. Disconnect the device from the all networking ports and connect it to another isolated network; the device changes its HA state to "Standlalone", displayed in the Web interface 'High Availability' field.

Connect to the device's Web interface (OAMP address) and change the subnet of the Maintenance interface in the Multiple Interface table.

Configure the HA Remote Address of the Active device (B) to correspond with the new subnet.

Reset the device.

Check that your settings were successfully applied.

Disconnect the Active device (B) from all networking ports.

Re-connect the Redundant device (A) to the network; it now becomes the active device.

On device B:

a. Connect the device to another isolated network; the device changes its HA state to "Standlalone", displayed in the Web interface 'High Availability' field.

Connect to the device's Web interface (OAMP address) and change the subnet of the Maintenance interface in the Multiple Interface table.

Configure the HA Remote Address (i.e., IP address of device A).

Reset the device.

Check that your settings were successfully applied.

Re-connect device B to the network; the Web interface 'High Availability' field now displays "Operational".

21.3 Configuring Firewall Allowed Rules

The device allows you to add firewall rules that can deny or allow specified traffic. This is done in the Firewall Settings page, as described later in this manual (see Configuring Firewall Settings). If you do add firewall rules that block certain traffic, you then also need to add certain rules that ensure that traffic concerned with the HA feature is allowed. These allowed HA rules include the following:

Keep-alive packets between the Active and Redundant devices - for example, Rules #1 and #2 in the figure below.

HA control and data packets between the Active and Redundant devices - for example, Rules #3 and #4 in the figure below.

HA control and data packets between the Active and Redundant devices after a switch over - for example, Rules #5 and #6 in the figure below. These rules are the same as Rules #3 and #4 respectively, but are required as the TCP source and destination port IDs are not symmetric.

HTTP protocol for file transferring - Rule #7 in the figure below.

HTTP protocol for file transferring after switch over - for example, Rule #8 (same as Rule #7) in the figure below.

The figure below displays an example of the required firewall rules. In this example, 10.31.4.61 is the HA Maintenance interface of the Redundant device, and 10.31.4.62 is the HA Maintenance interface of the Active device. "HA_IF" is the name of the Maintenance interface.

Figure 21-1: Allowed Firewall Rules for HA

Edit Rule	Rule Status	Source IP	Source Port	Prefix Length	Local Port Range	Protocol	Use Specific Interface	Interface Name	Packet Size	Byte rate	Burst Bytes	Action Upon Match	Match Count
0	Active	0.0.0.0	0	0	80-80	tcp	Enable	O+M+C	0	0	0	ALLOW	248
1	Active	10.31.4.61	669	32	669-669	udp	Enable	HA_IF	0	0	0	ALLOW	921
2	Active	10.31.4.62	669	32	669-669	udp	Enable	HA_IF	0	0	0	ALLOW	0
3	Active	10.31.4.61	0	32	2442-2442	TCP	Enable	HA_IF	0	0	0	ALLOW	57
4	Active	10.31.4.62	2442	32	0-65535	TCP	Enable	HA_IF	0	0	0	ALLOW	0
5	Active	10.31.4.61	2442	32	0-65535	TCP	Enable	HA_IF	0	0	0	ALLOW	0
6	Active	10.31.4.62	0	32	2442-2442	TCP	Enable	HA_IF	0	0	0	ALLOW	0
7	Active	10.31.4.61	80	32	0-65535	TCP	Enable	HA_IF	0	0	0	ALLOW	0
8	Active	10.31.4.62	80	32	0-65535	TCP	Enable	HA_IF	0	0	0	ALLOW	0
9	Not Active	0.0.0.0	0	0	0 - 65535	Any	Disable	None	0	0	0	Block	0

22 Troubleshooting HA

The table below provides troubleshooting for HA.

Table 22-1: Troubleshooting HA

Problem	During initial installation, the device fails to load in HA mode (indicated in the Home page by the 'High Availability' field displaying an empty string or an error status)
Corrective Actions	Verify that the HA feature key is enabled and installed on the device. Ensure that a valid Maintenance network interface has been defined. Ensure that the 'HA Remote Address' parameter is set with an address that is in the same subnet as the Maintenance interface address.
Problem	Both devices load in HA Active state
Corrective Actions	1 Verify network connectivity of each device with the Maintenance interface. Ensure on both devices that the 'Underlying Interface' value (i.e., Ethernet port group) of the Maintenance interface is configured correctly.

23 HA Maintenance

This section describes HA maintenance procedures.

23.1 Maintenance of the Redundant Device

As described before, the only interface that is operational on the Redundant device is the Maintenance interface. For maintenance, there are several protocols available on this interface (unlike the Active device which use the logical OAMP / management interface for these protocols):

Syslog: To receive Syslog messages from the Redundant device, ensure there is a valid VLAN and route defined from the system maintenance network to where the Syslog server is located on the network.

Telnet: A Telnet server is always available on the Redundant device (even if it is disabled by configuration)

23.2 Replacing a Failed Device

If you need to replace a non-functional device with a new one, the new device must be configured as the second device, as described in 'Stage 2: Configure the Second Device' on page 234. This new device must be configured with the same HA configuration as was done on the replaced device.

23.3 Forcing a Switchover

If required, you can force a switchover between Active and Redundant SBCs. For more information, see 'High Availability Maintenance' on page 246.

23.4 Software Upgrade

The following types of software upgrades are available on the HA system:

Software Upgrade with Device Reset: Both Active and Redundant devices burn and reboot with the new software version. This method is quick and simple, but it does not maintain service (i.e., traffic affecting).

Hitless Software Upgrade: This method maintains service (i.e., not traffic affecting). The process takes longer than the method above and can be summarized as follows:

a. The Redundant device burns and resets with the new software version.

A switch over is done between the Active and Redundant devices, whereby the Redundant device becomes the active one.

The previously Active device burns and resets with the new software version.

The previously Active device switches back to being active.

For more information on upgrading the software, see 'Software Upgrade Wizard' on page 252.

Part VI

Maintenance

This part describes the maintenance procedures.

24 Basic Maintenance

The Maintenance Actions page allows you to perform the following:

Reset the device - see 'Resetting the Device' on page 242

Lock and unlock the device - see 'Locking and Unlocking the Device' on page 244

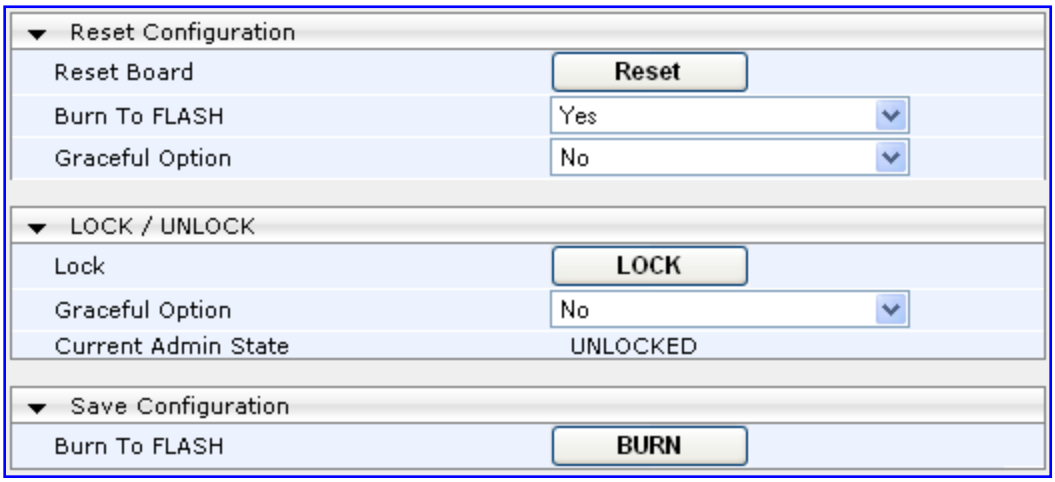
Save configuration to the device's flash memory - see 'Saving Configuration' on page 245

To access the Maintenance Actions page, do one of the following:

On the toolbar, click the **Device Actions** button, and then from the drop-down menu, choose **Reset**.

On the Navigation bar, click the **Maintenance** tab, and then in the Navigation tree, select the **Maintenance** menu and choose **Maintenance Actions**.

Figure 24-1: Maintenance Actions Page



24.1 Resetting the Device

The Maintenance Actions page allows you to remotely reset the device. In addition, before resetting the device, you can choose the following options:

Save the device's current configuration to the device's flash memory (non-volatile).

Perform a graceful shutdown, i.e., device reset starts only after a user-defined time (i.e., timeout) or after no more active traffic exists (the earliest thereof).

Notes:

Throughout the Web interface, parameters preceded by the lightning symbol are not applied on-the-fly and require that you reset the device for them to take effect.

When you modify parameters that require a device reset, once you click the **Submit** button in the relevant page, the toolbar displays "Reset" (see 'Toolbar' on page 20) to indicate that a device reset is required.

After you reset the device, the Web GUI is displayed in Basic view (see 'Displaying Navigation Tree in Basic and Full View' on page 21).

Upon reboot, the device restores the settings from its configuration file. However, if reboot attempts fail three times consecutively, the device resets the configuration file by restoring factory defaults before attempting to reboot.

To reset the device:

1. Open the Maintenance Actions page (see 'Basic Maintenance' on page 241).

Under the 'Reset Configuration' group, from the 'Burn To FLASH' drop-down list, select one of the following options:

Yes: The device's current configuration is saved (*burned*) to the flash memory prior to reset (default).

No: Resets the device without saving the current configuration to flash (discards all unsaved modifications).

Under the 'Reset Configuration' group, from the 'Graceful Option' drop-down list, select one of the following options:

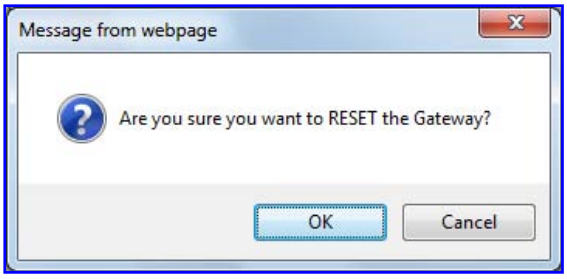
Yes: Reset starts only after the user-defined time in the 'Shutdown Timeout' field (see Step 4) expires or after no more active traffic exists (the earliest thereof). In addition, no new traffic is accepted.

No: Reset starts regardless of traffic, and any existing traffic is terminated at once.

In the 'Shutdown Timeout' field (relevant only if the 'Graceful Option' in the previous step is set to **Yes**), enter the time after which the device resets. Note that if no traffic exists and the time has not yet expired, the device resets.

Click the **Reset** button; a confirmation message box appears, requesting you to confirm.

Figure 24-2: Reset Confirmation Message Box



Click **OK** to confirm device reset; if the parameter 'Graceful Option' is set to **Yes** (in Step 3), the

reset is delayed and a screen displaying the number of remaining calls and time is displayed. When the device begins to reset, a message appears notifying you of this.

24.2 Locking and Unlocking the Device

The Lock and Unlock options allow you to lock the device so that it doesn't accept any new calls. This is useful when, for example, you are uploading new software files to the device and you don't want any traffic to interfere with the process.

To lock the device:

1. Open the Maintenance Actions page (see 'Basic Maintenance' on page 241).

Under the 'LOCK / UNLOCK' group, from the 'Graceful Option' drop-down list, select one of the following options:

Yes: The device is 'locked' only after the user-defined time in the 'Lock Timeout' field (see Step 3) expires or no more active traffic exists (the earliest thereof). In addition, no new traffic is accepted.

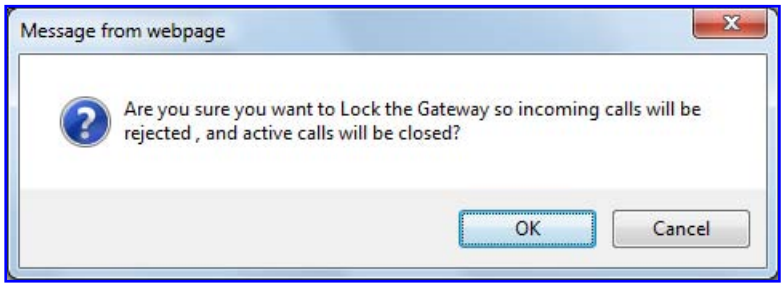
No: The device is 'locked' regardless of traffic. Any existing traffic is terminated immediately.

Note: These options are only available if the current status of the device is in the Unlock state.

In the 'Lock Timeout' field (relevant only if the parameter 'Graceful Option' in the previous step is set to **Yes**), enter the time (in seconds) after which the device locks. Note that if no traffic exists and the time has not yet expired, the device locks.

Click the **LOCK** button; a confirmation message box appears requesting you to confirm device Lock.

Figure 24-3: Device Lock Confirmation Message Box


Message from webpage
Are you sure you want to Lock the Gateway so incoming calls will be rejected , and active calls will be closed?
OK
Cancel

Click **OK** to confirm device Lock; if 'Graceful Option' is set to **Yes**, the lock is delayed and a screen displaying the number of remaining calls and time is displayed. Otherwise, the lock process begins immediately. The Current Admin State' field displays the current state - "LOCKED" or "UNLOCKED".

To unlock the device:

1. Open the Maintenance Actions page (see 'Maintenance Actions' on page 241).

Under the 'LOCK / UNLOCK' group, click the **UNLOCK** button. Unlock starts immediately and the device accepts new incoming calls.

Note: The Home page's General Information pane displays whether the device is locked or unlocked (see 'Using the Home Page' on page 36).

24.3 Saving Configuration

The Maintenance Actions page allows you to save (*burn*) the current parameter configuration (including loaded auxiliary files) to the device's *non-volatile* memory (i.e., flash). The parameter modifications that you make throughout the Web interface's pages are temporarily saved (to the *volatile* memory - RAM) when you click the **Submit** button on these pages. Parameter settings that are saved only to the device's RAM revert to their previous settings after a hardware/software reset (or power failure). Therefore, to ensure that your configuration changes are retained, you must save them to the device's flash memory using the burn option described below.

To save the changes to the non-volatile flash memory :

1. Open the Maintenance Actions page (see 'Basic Maintenance' on page 241).

Under the 'Save Configuration' group, click the **BURN** button; a confirmation message appears when the configuration successfully saves.

Notes:

Saving configuration to the *non-volatile* memory may disrupt current traffic on the device. To avoid this, disable all new traffic before saving, by performing a graceful lock (see 'Locking and Unlocking the Device' on page 244).

Throughout the Web interface, parameters preceded by the lightning symbol are not applied on-the-fly and require that you reset the device for them to take effect (see 'Resetting the Device' on page 242).

The Home page's General Information pane displays whether the device is currently "burning" the configuration (see 'Using the Home Page' on page 36).

25 High Availability Maintenance

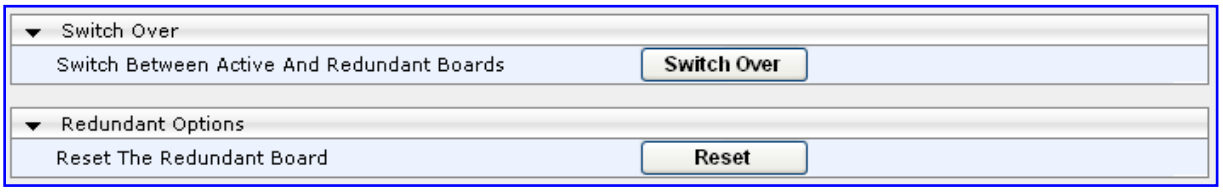
The High Availability Maintenance page allows you to perform a switch-over between the Active and Redundant SBCs. It also allows you to reset the Redundant SBC.

Note: When performing a switchover or a reset on the Redundant SBC, the HA mode becomes temporarily unavailable.

To perform an SBC switch-over:

2. Open the High Availability Maintenance page (**Maintenance** tab >**Maintenance** menu >**High Availability Maintenance**).

Figure 25-1: High Availability Maintenance Page



Under the 'Switch Over' group, click **Switch Over**; a confirmation box appears requesting you to confirm.

Click **OK**.

To reset the Redundant SBC:

3. Under the 'Redundant Options' group, click **Reset**; a confirmation box appears requesting you to confirm.

Click **OK**.

26 Software Upgrade

The **Software Update** menu allows you to upgrade the device's software, install Software Upgrade Key, and load/save configuration file. This menu includes the following page items:

Load Auxiliary Files (see 'Loading Auxiliary Files' on page 247)

Software Upgrade Key (see 'Loading Software Upgrade Key' on page 250)

Software Upgrade Wizard (see 'Software Upgrade Wizard' on page 252)

Configuration File (see 'Backing Up and Loading Configuration File' on page 255)

26.1 Loading Auxiliary Files

The Load Auxiliary Files page allows you to load various auxiliary files to the device. These auxiliary files are briefly described in the table below:

Table 26-1: Auxiliary Files Descriptions

File	Description
INI	Provisions the device's parameters. The Web interface enables practically full device provisioning, but customers may occasionally require new feature configuration parameters in which case this file is loaded. **Note:** Loading this file only provisions those parameters that are included in the *ini* file. For more information on the *ini* file, see 'INI File-Based Management' on page 53.
Dial Plan	This file contains dialing plans, used by the device. For more information on the Dial Plan file, see Dial Plan File on page 249.
User Info	The User Information file maps PBX extensions to IP numbers. This file can be used to represent PBX extensions as IP phones in the global 'IP world'. For more information on the User Info file, see 'User Information File' on page 249.

Notes:

- You can schedule automatic loading of updated auxiliary files using HTTP/HTTPS (for more information, refer to the *Product Reference Manual*).
- For more information on auxiliary files, see 'Auxiliary Configuration Files' on page 249.
- When loading an *ini* file using this Web page, parameters that are excluded from the loaded *ini* file retain their current settings (*incremental*).
- Saving an auxiliary file to flash memory may disrupt traffic on the device. To avoid this, disable all traffic on the device, by performing a graceful lock (see 'Locking and Unlocking the Device' on page 244).
- For deleting auxiliary files, see 'Viewing Device Information' on page 262.

The auxiliary files can be loaded to the device using the Web interface's 'Load Auxiliary Files page, as described in the procedure below.

To load an auxiliary file to the device using the Web interface:

1. Open the Load Auxiliary Files page (**Maintenance** tab >**Software Update** menu >**Load Auxiliary Files**).

Figure 26-1: Load Auxiliary Files Page

Note: The appearance of certain file load fields depends on the installed Software Upgrade Key.

Click the **Browse** button corresponding to the file type that you want to load, navigate to the folder in which the file is located, and then click **Open**; the name and path of the file appear in the field next to the **Browse** button.

Click the **Load File** button corresponding to the file you want to load.

Repeat steps 2 through 3 for each file you want to load.

Save the loaded auxiliary files to flash memory, see 'Saving Configuration' on page 245and reset the device (if you have loaded a Call Progress Tones file), see 'Resetting the Device' on page 242.

26.1.1 Auxiliary Configuration Files

This section describes the auxiliary files that can be loaded to the device:

User Information (see 'User Information File' on page 249)

You can load these auxiliary files to the device using one of the following methods:

Loading the files directly to the device using the device's Web interface (see 'Loading Auxiliary Files' on page 247).

Specifying the auxiliary file name in the *ini* file (see 'Auxiliary and Configuration Files Parameters' on page 358) and then loading the *ini* file to the device. The Auxiliary files listed in the *ini* file are then uploaded to the device through TFTP during device startup. If the *ini* file does not contain a specific auxiliary file type, the device uses the last auxiliary file of that type that was stored on its non-volatile memory.

26.1.1.1 User Information Filefor SBC Users Database

The User Information file can be loaded to the device by using one of the following methods:

ini file, using the parameter UserInfoFileName (described in 'Auxiliary and Configuration Files Parameters' on page 358)

Web interface (see 'Loading Auxiliary Files' on page 247)

Automatic update mechanism, using the parameter UserInfoFileURL (refer to the *Product Reference Manual*)

You can create an SBC users database from a loaded User Information file. This feature is enabled by the EnableUserInfoUsage parameter. Up to 600 SBC users can be defined in the User Information file.

The User Information file lists the SBC users under the "**[SBC]"** section, as shown below:

```
[ SBC ]
FORMAT LocalUser,UserName,Password,IPGroupID
john,john_user,john_pass,2
sue,sue_user,sue_pass,1
```

where:

LocalUser identifies the user and is used as the URI user part for the AOR in the database

UserName is the user's authentication username.

Password is the user's authentication password.

IPGroupID is the IP Group ID to which the user belongs and is used as the URI source host part for the AOR in the database

The SBC users database can be used for the following:

Register to an external registrar server on behalf of a specific user.

Authenticate (for any SIP request and as a client) on behalf of a specific user if challenged by an external server.

Authenticate (as a server) incoming user requests (for SBC security).

If the SBC registers on behalf of users and the users do not perform registration at all, any SIP request destined to the user is routed to the Proxy Set associated with the user's IP Group.

26.2 Loading Software Upgrade Key

The Software Upgrade Key Status page allows you to load a new Software Upgrade Key to the device. The device is supplied with a Software Upgrade Key, which determines the device's supported features, capabilities, and available resources. The availability of certain Web pages depends on the loaded Software Upgrade Key. You can upgrade or change your device's supported features by purchasing a new Software Upgrade Key to match your requirements.

The Software Upgrade Key is provided in string format in a text-based file (.out). When you load a Software Upgrade Key, it is loaded to the device's non-volatile flash memory and overwrites the previously installed key.

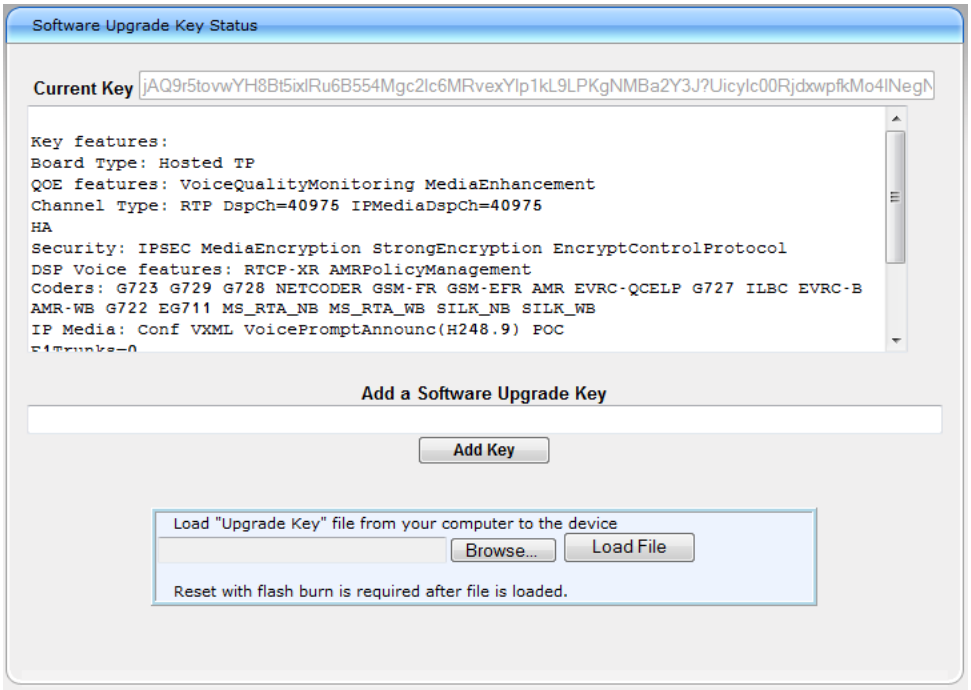
Warning: Do not modify the contents of the Software Upgrade Key file.

Note: The Software Upgrade Key is an encrypted key.

To load a Software Upgrade Key:

1. Open the Software Upgrade Key Status page (**Maintenance** tab >**Software Update** menu >**Software Upgrade Key**).

Figure 26-2: Software Upgrade Key Status Page


Software Upgrade Key Status
Current Key
jAQ9r5tovwYH8Bt5ixlRu6B554Mgc2lc6MRvexYlp1kL9LPKgNMBa2Y3J?UicyIc00RjdxwpfkMo4lNegN
Key features:
Board Type: Hosted TP
QOE features: VoiceQualityMonitoring MediaEnhancement
Channel Type: RTP DspCh=40975 IPMediaDspCh=40975
HA
Security: IPSEC MediaEncryption StrongEncryption EncryptControlProtocol
DSP Voice features: RTCP-XR AMRPolicyManagement
Coders: G723 G729 G728 NETCODER GSM-FR GSM-EFR AMR EVRC-QCELP G727 ILBC EVRC-B
AMR-WB G722 EG711 MS_RTA_NB MS_RTA_WB SILK_NB SILK_WB
IP Media: Conf VXML VoicePromptAnnounc(H248.9) POC
Add a Software Upgrade Key
Add Key
Load "Upgrade Key" file from your computer to the device
Browse...
Load File
Reset with flash burn is required after file is loaded.

Backup your current Software Upgrade Key as a precaution so that you can re-load this backup key to restore the device's original capabilities if the new key doesn't comply with your requirements:

a. In the 'Current Key' field, copy the string of text and paste it into any standard text file.

Save the text file to a folder on your PC with a name of your choosing and file extension .out.

Open the new Software Upgrade Key file and ensure that the first line displays '**[LicenseKeys]**' and that it contains one or more lines in the following format: S/N<serial number> = <long Software Upgrade Key string>

For example: S/N370604 = jCx6r5tovCIKaBBbhPtT53Yj...

Follow one of the following procedures, depending on whether you are loading a single or multiple key S/N lines:

Single key S/N line(applicable only to non-HA system)**:**

a. Open the Software Upgrade Key text file (using, for example, Microsoft® Notepad).

Select and copy the key string and paste it into the field 'Add a Software Upgrade Key'.

Click the **Add Key** button.

Multiple S/N lines (as shown below):

Figure 26-3: Software Upgrade Key with Multiple S/N Lines

a. In the 'Load Upgrade Key file' field, click the **Browse** button and navigate to the folder in which the Software Upgrade Key text file is located on your PC.

Click the **Load File** button; the new key is loaded to the device and validated. If the key is valid, it is burned to memory and displayed in the 'Current Key' field.

Verify that the Software Upgrade Key file was successfully loaded to the device, by using one of the following methods:

In the 'Key features' group, ensure that the features and capabilities activated by the installed string match those that were ordered.

Access the Syslog server (refer to the *Product Reference Manual*) and ensure that the following message appears in the Syslog server:
"S/N___ Key Was Updated. The Board Needs to be Reloaded with ini file\n".

Reset the device; the new capabilities and resources are active.

Note: If the Syslog server indicates that the Software Upgrade Key file was unsuccessfully loaded (i.e., the 'SN_' line is blank), do the following preliminary troubleshooting procedures:

1. Open the Software Upgrade Key file and check that the S/N line appears. If it does not appear, contact your representative.

Verify that you've loaded the correct file. Open the file and ensure that the first line displays **[LicenseKeys].**

Verify that the content of the file has not been altered.

26.3 Software Upgrade Wizard

The Software Upgrade Wizard allows you to upgrade the device's firmware (compressed .cmp file) as well as load an *ini* file (typically loaded using the Load Auxiliary File page described in 'Loading Auxiliary Files' on page 247). However, it is mandatory when using the wizard to first load a .cmp file to the device. You can then choose to also load an *ini* file, but this cannot be done without first loading a .cmp file. For the *ini* file type, you can choose to load a new file or not load a file but use the existing file (i.e., maintain existing configuration) running on the device.

Warning: The Software Upgrade Wizard requires the device to be reset at the end of the process, which may disrupt traffic. To avoid this, disable all traffic on the device before initiating the wizard, by performing a graceful lock (see 'Basic Maintenance' on page 241).

Notes:

- Before upgrading the device, it is recommended that you save a copy of the device's configuration settings (i.e., *ini* file) to your PC. If an upgrade failure occurs, you can then restore your configuration settings by uploading the backup file to the device. For saving and restoring configuration, see 'Backing Up and Loading Configuration File' on page 255.
- Before you can load an *ini*, you must first load a .cmp file.
- When you activate the wizard, the rest of the Web interface is unavailable. After the files are successfully loaded, access to the full Web interface is restored.
- If you upgraded your .cmp and the "SW version mismatch" message appears in the Syslog or Web interface, then your Software Upgrade Key does not support the new .cmp file version. Contact support for assistance.
- If you use the wizard to load an *ini* file, parameters excluded from the *ini* file are assigned default values (according to the .cmp file running on the device), thereby, overriding values previously defined for these parameters.
- You can schedule automatic loading of these files using HTTP/HTTPS (refer to the *Product Reference Manual*).

To load files using the Software Upgrade Wizard:

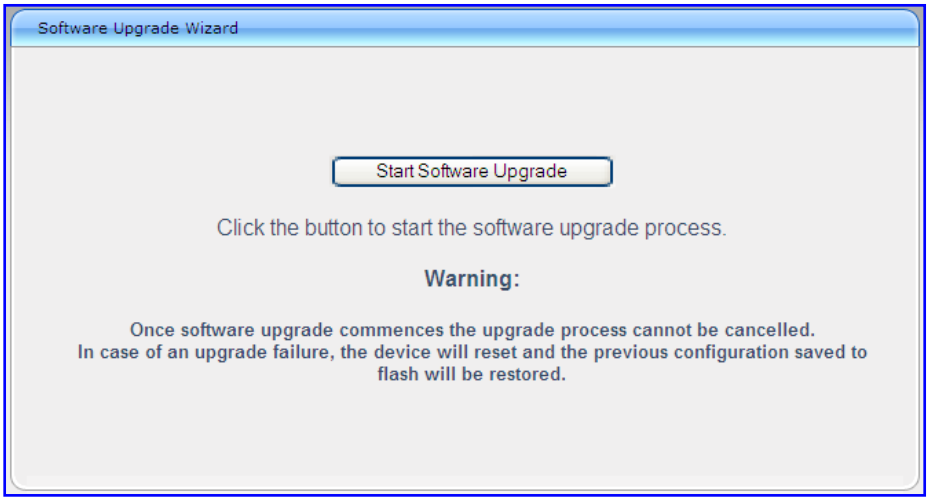
1. Stop all traffic on the device using the Graceful Lock feature (refer to the warning bulletin above).

Open the Software Upgrade wizard, by performing one of the following:

Select the **Maintenance** tab, click the **Software Update** menu, and then click **Software Upgrade Wizard**.

On the toolbar, click **Device Actions**, and then choose **Software Upgrade Wizard**.

Figure 26-4: Start Software Upgrade Wizard Screen


Software Upgrade Wizard
Start Software Upgrade
Click the button to start the software upgrade process.
Warning:
Once software upgrade commences the upgrade process cannot be cancelled.
In case of an upgrade failure, the device will reset and the previous configuration saved to flash will be restored.

Click the **Start Software Upgrade** button; the wizard starts, requesting you to browses to a .cmp file for uploading.

Note: At this stage, you can quit the Software Update Wizard, by clicking **Cancel**, without requiring a device reset. However, once you start uploading a cmp file, the process must be completed with a device reset. If you choose to quit the process in any of the subsequent pages, the device resets.

Click the **Browse** button, navigate to the .cmp file, and then click **Load File**; a progress bar appears displaying the status of the loading process. When the .cmp file is successfully loaded to the device, a message appears notifying you of this.

If you want to load **only** a .cmp file, then click the **Reset** button to reset the device with the newly loaded .cmp file, utilizing the existing configuration (*ini*) file. To load additional files, skip to Step 7.

Note: Device reset may take a few minutes depending on cmp file version (this may even take up to 10 minutes).

Click the **Next** button; the wizard page for loading an *ini* file appears. You can now perform one of the following:

Load a new *ini* file: Click **Browse**, navigate to the *ini* file, and then click **Send File**; the *ini* file is loaded to the device and you're notified as to a successful loading.

Retain the existing configuration (*ini* file): Do not select an *ini* file, and ensure that the 'Use existing configuration' check box is selected (default).

Return the device's configuration settings to factory defaults: Do not select an *ini* file, and clear the 'Use existing configuration' check box.

When you have completed loading all the desired files, click the **Next** button until the last wizard page appears ("FINISH" is highlighted in the left pane).

Click the **Reset** button to complete the upgrade process; the device 'burns' the newly loaded files to flash memory and then resets the device.

Note: Device reset may take a few minutes (depending on .cmp file version, this may even take up to 30 minutes).

After the device resets, the End of Process wizard page appears displaying the new .cmp and auxiliary files loaded to the device.

Figure 26-5: End Process Wizard Page

Click **End Process** to close the wizard; the Web Login dialog box appears.

Enter your login user name and password, and then click **OK**; a message box appears informing you of the new .cmp file.

Click **OK**; the Web interface becomes active, reflecting the upgraded device.

26.4 Backing Up and Loading Configuration File

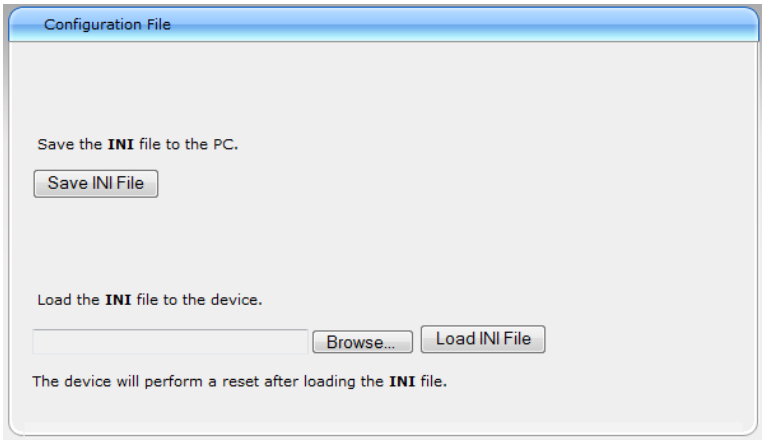
You can save a copy/backup of the device's current configuration settings as an *ini* file to a folder on your PC, using the 'Configuration File page. The saved *ini* file includes only parameters that were modified and parameters with other than default values. The Configuration File page also allows you to load an *ini* file to the device. If the device has "lost" its configuration, you can restore the device's configuration by loading the previously saved *ini* file or by simply loading a newly created *ini* file.

Note: When loading an *ini* file using this Web page, parameters not included in the *ini* file are reset to default settings.

To save the ini / data file:

Open the Configuration File page (**Maintenance** tab >**Software Update** menu >**Configuration File**). You can also access this page from the toolbar, by clicking **Device Actions**, and then choosing **Load Configuration File** or **Save Configuration File**.

Figure 26-6: Configuration File Page


Configuration File
Save the INI file to the PC.
Save INI File
Load the INI file to the device.
Browse...
Load INI File
The device will perform a reset after loading the INI file.

To save the Voice *ini* file to a folder on your PC:

2. Click the **Save INI File** button; the 'File Download' dialog box appears.

Click the **Save** button, navigate to the folder in which you want to save the *ini* file on your PC, and then click **Save**; the device copies the *ini* file to the selected folder.

To load the ini file:

3. Click the **Browse** button, navigate to the folder in which the *ini* file is located, select the file, and then click **Open**; the name and path of the file appear in the field beside the **Browse** button.

Click the **Load INI File** button, and then at the prompt, click **OK**; the device uploads the *ini* file and then resets (from the *cmp* version stored on the flash memory). Once complete, the Login screen appears, requesting you to enter your user name and password.

27 System Snapshot

System Snapshot captures a complete OpenTouch™ Session Border Controller state, including the following:

Installed OpenTouch™ Session Border Controller software

Current configuration

Auxiliary files

Software Feature Key

The device does a first snapshot automatically upon initial installation. You may do up to 10 additional snapshots, if required, as described in Section 27.1 below.You can restore OpenTouch™ Session Border Controller to a previous snapshot, as described in Section 27.2.

27.1 Taking a Snapshot

The procedure below describes how to make a snapshot of the current device state.

To take a snapshot,using the CLI:

4. Establish a CLI connection with the device.

5. At the prompt, type the following command:

   #**enable**

6. At the prompt, type the password:

   Password: **Admin**

7. At the prompt, type the following commands to take a snapshot:

   # **configure system**
   # **startup-n-recovery**
   # **create-system-snapshot** `<snapshot name>`

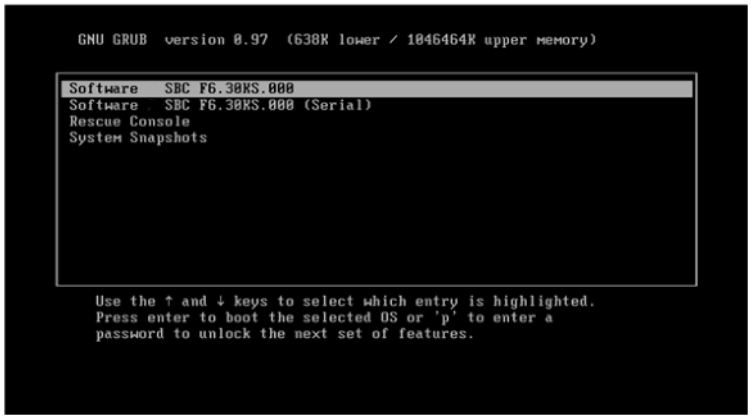
27.2 Returning to a Snapshot State

If you want to restore the device to a previous snapshot state, then follow the procedure below.

To return to a previous snapshot state:

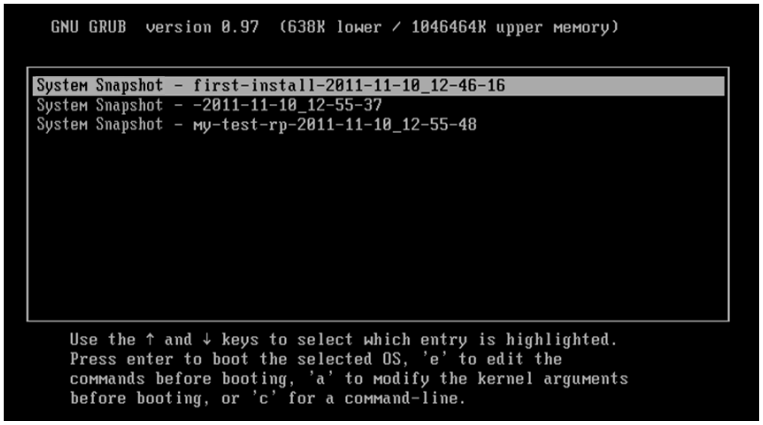
8. Reboot the server.
9. In the GRUB menu, displayed for 5 seconds during the server start-up, press the Down ↓ key to prevent the server from starting the OpenTouch™ Session Border Controller software.

Figure 27-1: GRUB Menu


GNU GRUB version 0.97 (638K lower / 1046464K upper memory)
Software SBC F6.30KS.000
Software SBC F6.30KS.000 (Serial)
Rescue Console
System Snapshots
Use the ↑ and ↓ keys to select which entry is highlighted.
Press enter to boot the selected OS or 'p' to enter a
password to unlock the next set of features.

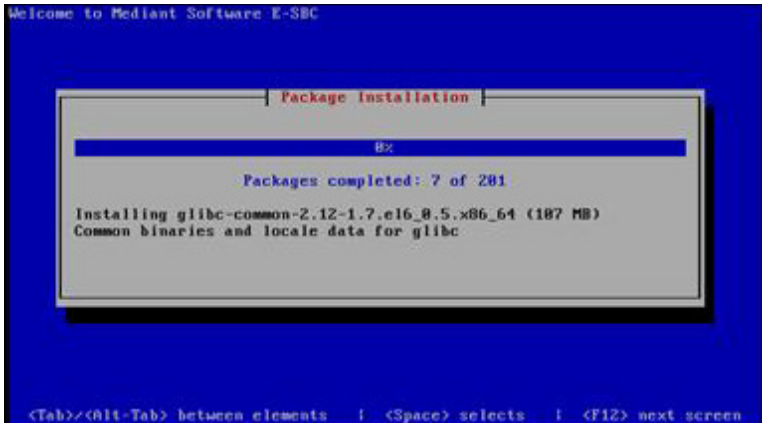
10. Select **System Snapshots** and then press Enter; you're prompted to select a snapshot.

Figure 27-2: Selecting a Snapshot


GNU GRUB version 0.97 (638K lower / 1046464K upper memory)
System Snapshot - first-install-2011-11-10_12-46-16
System Snapshot - -2011-11-10_12-55-37
System Snapshot - my-test-rp-2011-11-10_12-55-48
Use the ↑ and ↓ keys to select which entry is highlighted.
Press enter to boot the selected OS, 'e' to edit the
commands before booting, 'a' to modify the kernel arguments
before booting, or 'c' for a command-line.

11. Select a snapshot and then press Enter; the system returns to the selected snapshot state.

Figure 27-3: System Returning to Snapshot State


Welcome to Mediant Software E-SBC
Package Installation
Packages completed: 7 of 281
Installing glibc-common-2.12-1.7.el6_0.5.x86_64 (107 MB)
Common binaries and locale data for glibc
<Tab>/<Alt-Tab> between elements | <Space> selects | <F12> next screen

This operation may take up to 10 minutes to complete. The system automatically reboots after the return is complete.

28 Restoring Factory Default Settings

You can restore the device's configuration to factory defaults using one of the following methods:

Using the CLI (see 'Restoring Defaults using CLI' on page 259)

Loading an empty *ini* file (see 'Restoring Defaults using an ini File' on page 260)

28.1 Restoring Defaults using CLI

The device can be restored to factory defaults using CLI, as described in the procedure below.

To restore factory defaults using CLI:

1. Access the CLI:
   a. Connect the RS-232 serial port of the device to the communication port on your PC. For cabling the device, refer to the *Hardware Installation Manual*.

   Establish serial communication with the device using a serial communication program (such as HyperTerminal™) with the following communication port settings:

   Baud Rate:115,200 bps

   Data Bits: 8

   Parity: None

   Stop Bits: 1

   Flow Control: None

At the CLI prompt, type the username (default is "Admin" - case sensitive), and then press Enter:

#Username: **Admin**

At the prompt, type the password (default is "Admin" - case sensitive), and then press Enter:

#Password: **Admin**

At the prompt, type the following, and then press Enter:

#**enable**

At the prompt, type the password again, and then press Enter:

#Password: **Admin**

At the prompt, type the following to reset the device to default settings, and then press Enter:

write factory

28.2 Restoring Defaults using an ini File

You can restore the device to factory default settings by loading an empty *ini* file to the device, using the Web interface's Configuration File page (see 'Backing Up and Loading Configuration File' on page 255). The only settings that are not restored to default are the management (OAMP) LAN IP address and the Web interface's login user name and password. The loaded *ini* file must be empty (i.e., contain no parameters), or include only comment signs (i.e., semicolons ";") preceding lines (parameters). The default values assigned to the parameters are according to the *cmp* file running on the device.

Part VII

Status, Performance Monitoring & Reporting

This part describes how to view the status of the device,monitor its performance, and report performance information to external application servers.

29 System Status

This section describes how to view system status.

Device information - see 'Viewing Device Information' on page 262

Ethernet port information - see 'Viewing Ethernet Port Information' on page 263

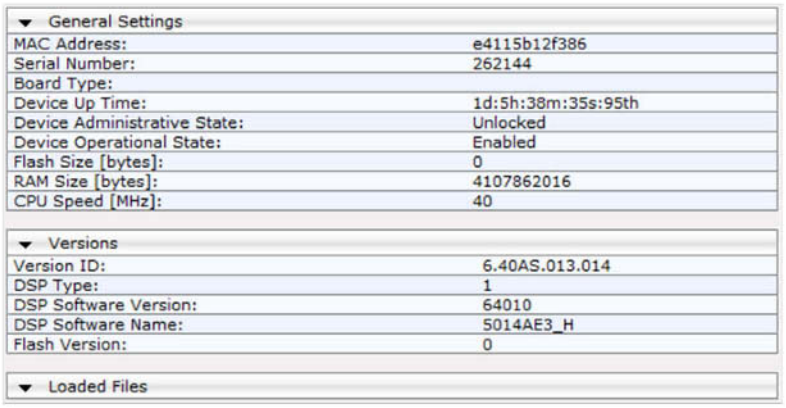
29.1 Viewing Device Information

The Device Information page displays the device's specific hardware and software product information. This information can help you expedite troubleshooting. Capture the page and e-mail it to Alcatel-Lucent Technical Support personnel to ensure quick diagnosis and effective corrective action. This page also displays any loaded files used by the device (stored in the RAM) and allows you to remove them.

To access the Device Information page:

Open the Device Information page (**Status & Diagnostics** tab >**System Status** menu >**Device Information**).

Figure 29-1: Device Information Page



To delete a loaded file:

Click the **Delete** button corresponding to the file that you want to delete. Deleting a file takes effect only after device reset (see 'Resetting the Device' on page 242).

29.2 Viewing Ethernet Port Information

The Ethernet Port Information page displays read-only information on the Ethernet port connections. This includes information such as activity status, duplex mode, and speed.

Note: The Ethernet Port Information page can also be accessed from the Home page (see 'Using the Home Page' on page 36).

To view Ethernet port information:

Open the Ethernet Port Information page (**Status & Diagnostics** tab >**System Status** menu >**Ethernet Port Information**).

Figure 29-2: Ethernet Port Information Page

	Active	Speed	Duplex Mode	State
Group no.1				
1	Yes	100 Mbps	Half Duplex	Forwarding
2	Yes	100 Mbps	Half Duplex	Forwarding

Table 29-1: Ethernet Port Information Parameters

Parameter	Description
Active	Displays whether the port is active or not.
Speed	Displays the speed (in Mbps) of the Ethernet port.
Duplex Mode	Displays whether the port is half- or full-duplex mode.
State	Displays one of the following: "Forwarding" - port is receiving and sending data. "Disabled" - port is disabled

30 Carrier-Grade Alarms

This section describes how to view the following types of alarms:

Active alarms - see 'Viewing Active Alarms' on page 264

Alarm history - see 'Viewing Alarm History' on page 265

30.1 Viewing Active Alarms

The Active Alarms page displays a list of currently active alarms. You can also access this page from the Home page (see 'Using the Home Page' on page 36).

To view the list of active alarms:

Open the Active Alarms page (**Status & Diagnostics** tab >**System Status** menu >**Carrier-Grade Alarms**>**Active Alarms**).

Figure 30-1: Active Alarms Page

Sequential number	Severity	Source	Description	Date
1	Major	Chassis#0/PowerSupply#1	Power-Supply Alarm. Power-Supply is missing	26.10.2011 , 11:30:58
3	Minor	Board#1/EthernetLink#2	Ethernet link alarm. LAN port number 2 is down.	26.10.2011 , 11:31:3
4	Minor	Board#1/EthernetLink#4	Ethernet link alarm. LAN port number 4 is down.	26.10.2011 , 11:31:3
5	Minor	Board#1/EthernetLink#5	Ethernet link alarm. LAN port number 5 is down.	26.10.2011 , 11:31:3
6	Minor	Board#1/EthernetLink#8	Ethernet link alarm. LAN port number 8 is down.	26.10.2011 , 11:31:3
10	Major	Board#1	Configuration mismatch in the system. SYS_HA: Active and Redundant modules have different feature keys.	26.10.2011 , 11:36:37

For each alarm, the following information is provided:

Severity: severity level of the alarm:

Critical - alarm displayed in red

Major - alarm displayed in orange

Minor - alarm displayed in yellow

Source: unit from which the alarm was raised

Description: brief explanation of the alarm

Date: date and time that the alarm was generated

You can view the next 20 alarms (if exist), by clicking the **Go to page** button.

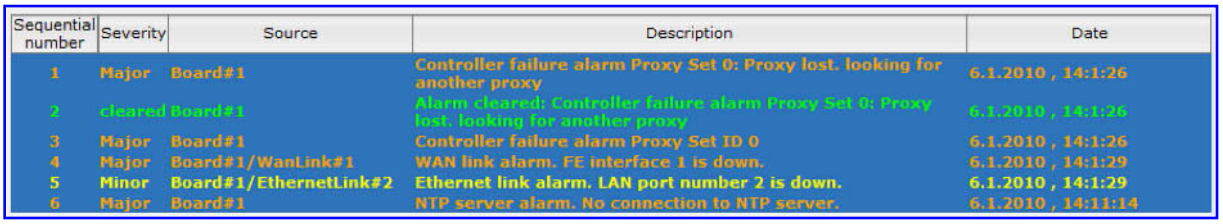
30.2 Viewing Alarm History

The Alarms History page displays a list of alarms that have been raised and traps that have been cleared.

To view the list of history alarms:

Open the Alarms History page (**Status & Diagnostics** tab >**System Status** menu >**Carrier-Grade Alarms**>**Alarms History**).

Figure 30-2: Alarms History Page



Sequential number	Severity	Source	Description	Date
1	Major	Board#1	Controller failure alarm Proxy Set 0: Proxy lost. looking for another proxy	6.1.2010 , 14:1:26
2	cleared	Board#1	Alarm cleared: Controller failure alarm Proxy Set 0: Proxy lost. looking for another proxy	6.1.2010 , 14:1:26
3	Major	Board#1	Controller failure alarm Proxy Set ID 0	6.1.2010 , 14:1:26
4	Major	Board#1/WanLink#1	WAN link alarm. FE interface 1 is down.	6.1.2010 , 14:1:29
5	Minor	Board#1/EthernetLink#2	Ethernet link alarm. LAN port number 2 is down.	6.1.2010 , 14:1:29
6	Major	Board#1	NTP server alarm. No connection to NTP server.	6.1.2010 , 14:11:14

For each alarm, the following information is provided:

Severity: severity level of the alarm:

Critical - alarm displayed in red

Major - alarm displayed in orange

Minor - alarm displayed in yellow

Cleared - alarm displayed in green

Source: unit from which the alarm was raised

Description: brief explanation of the alarm

Date: date and time that the alarm was generated

You can view the next 20 alarms (if exist), by clicking the **Go to page** button.

31 Performance Monitoring

This section describes how to view the following performance monitoring graphs:

Quality of Experience - see Viewing Quality of Experience on page 266

Average Call Duration - see Viewing Average Call Duration on page 268

31.1 Viewing Quality of Experience

The Quality Of Experience page provides statistical information on calls per SRD or IP Group. The statistics can be further filtered to display incoming and/or outgoing call direction, and type of SIP dialog (INVITE, SUBSCRIBE, or all).

Note: This page is available only if the SBC application has been enabled.

This page provides three pie charts:

Dialog Success Ratio: displays the SIP call and subscribe (SUBSCRIBE) dialog success-failed ratio.

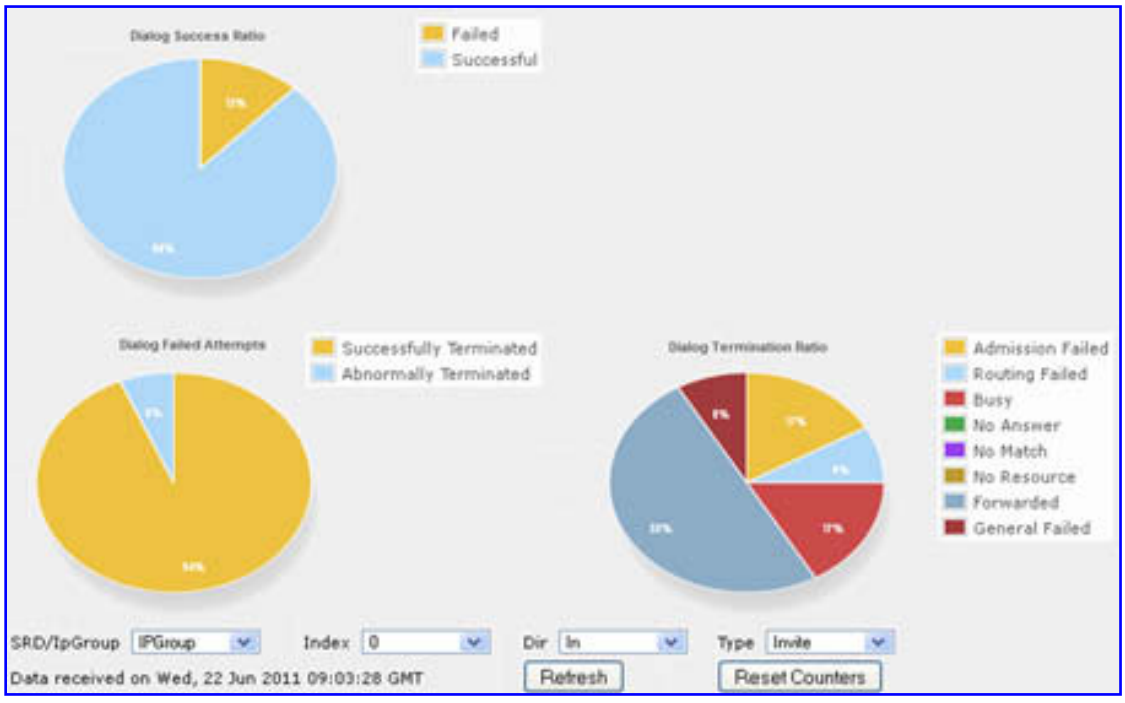
Dialog Failed Attempts: displays the failed call attempts. This includes the number of calls and subscribes which were successfully and abnormally terminated.

Dialog Termination Ratio: displays call termination by reason (e.g., due to no answer).

To view Quality of Experience:

2. Open the Quality Of Experience page (**Status & Diagnostics** tab >**Performance Monitoring** menu >**Quality Of Experience**).

Figure 31-1: Quality Of Experience Graph



From the 'SRD/IpGroup' drop-down list, select whether you want to view QoE for an SRD or IP Group.

From the 'Index' drop-down list, select the SRD or IP Group index.

From the 'Dir' drop-down list, select the call direction:

In - incoming calls

Out - outgoing calls

Both - incoming and outgoing calls

From the 'Type' drop-down list, select the SIP message type:

Invite - INVITE

Subscribe - SUBSCRIBE

Other - all SIP messages

To refresh the charts, click **Refresh**. To reset the counters, click **Reset Counters**.

31.2 Viewing Average Call Duration

The Average Call Duration page displays information about a specific SRD or IP Group. This page includes two graphs:

Upper graph: displays the number of calls (INVITEs).

Lower graph: displays the average call duration.

Note: This page is available only if the SBC application has been enabled.

To view average call duration:

3. Open the Average Call Duration page (**Status & Diagnostics** tab >**Performance Monitoring** menu >**Average Call Duration**).

Figure 31-2: Average Call Duration Graph

From the 'SRD/IpGroup' drop-down list, select whether you want to view information for an SRD or IP Group.

From the 'Index' drop-down list, select the SRD or IP Group index.

Use the **Zoom In** button to increase the displayed time resolution or the **Zoom Out** button to decrease it. Instead of using these zoom buttons, you can use the slide ruler. As you increase the resolution, more data is displayed on the graph. The minimum resolution is about 30 seconds; the maximum resolution is about an hour.

To pause the graph, click the **Pause** button; click **Play** to resume.

32 VoIP Status

This section describes how to view the following VoIP status and statistics:

IP network interface - see 'Viewing Active IP Interfaces' on page 269

Performance - see 'Viewing Performance Statistics' on page 270

SAS/SBC registered users - see Viewing SAS/SBC Registered Users on page 270

Call routing - see 'Viewing Call Routing Status' on page 271

Registration - see Viewing Registration Status on page 271

IP connectivity - see 'Viewing IP Connectivity' on page **Erreur ! Signet non défini.**

32.1 Viewing Active IP Interfaces

The IP Interface Status page displays the device's active IP interfaces, which are configured in the Multiple Interface Table page (see 'Configuring IP Interface Settings' on page 68).

To view the Active IP Interfaces page:

Open the IP Interface Status page (**Status & Diagnostics** tab >**VoIP Status** menu >**IP Interface Status**).

Figure 32-1: IP Interface Status Page

Index	Application Type	Address Type	Interface Mode	IP Address	Prefix Length	Gateway	VLAN ID	Interface Name
1	Maintenance	IPv4	IPv4 Manual	10.66.33.44	16	10.66.0.1	25	HA
0	O+M+C	IPv4	IPv4 Manual	10.3.50.40	16	10.3.0.1	1	O+M+C
2	Media	IPv4	IPv4 Manual	10.11.50.40	16	0.0.0.0	11	Media1
3	Media	IPv4	IPv4 Manual	10.13.50.40	16	0.0.0.0	13	Media2
4	Media	IPv4	IPv4 Manual	10.15.50.40	16	0.0.0.0	15	Media3
5	Media	IPv4	IPv4 Manual	10.12.50.40	16	0.0.0.0	112	Media4
6	Media	IPv4	IPv4 Manual	10.14.50.40	16	0.0.0.0	114	Media5

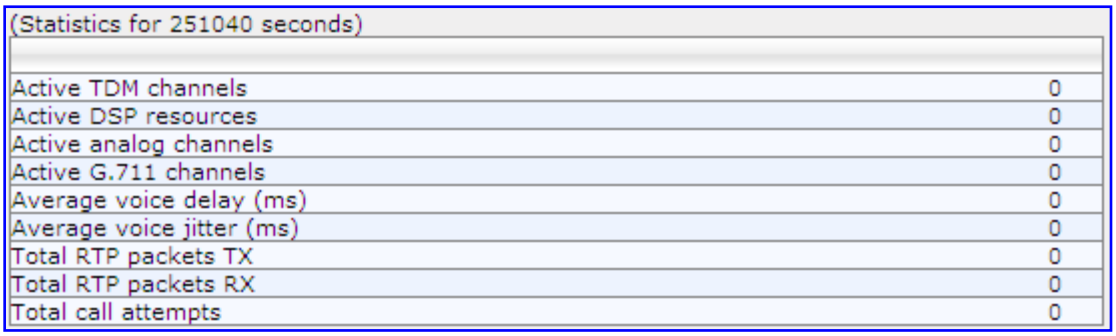
32.2 Viewing Performance Statistics

The Basic Statistics page provides read-only, device performance statistics. This page is refreshed every 60 seconds. The duration that the currently displayed statistics has been collected is displayed above the statistics table.

To view performance statistics:

Open the Basic Statistics page (**Status & Diagnostics** tab >**VoIP Status** menu >**Performance Statistics**).

Figure 32-2: Basic Statistics Page



(Statistics for 251040 seconds)	
Active TDM channels	0
Active DSP resources	0
Active analog channels	0
Active G.711 channels	0
Average voice delay (ms)	0
Average voice jitter (ms)	0
Total RTP packets TX	0
Total RTP packets RX	0
Total call attempts	0

To reset the performance statistics to zero, click the **Reset Statistics** button.

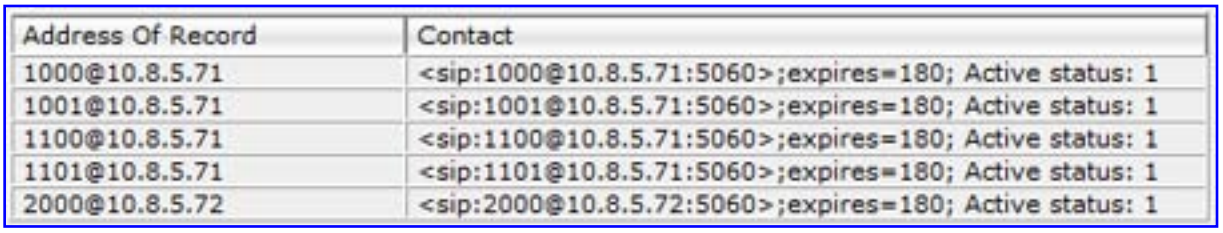
32.3 Viewing SAS/SBC Registered Users

The SAS/SBC Registered Users page displays a list of registered SAS/SBC users recorded in the device's database.

To view registered users:

Open the SAS/SBC Registered Users page (**Status & Diagnostics** tab >**VoIP Status** menu >**SAS/SBC Registered Users**).

Figure 32-3: SAS/SBC Registered Users Page



Address Of Record	Contact
1000@10.8.5.71	<sip:1000@10.8.5.71:5060>;expires=180; Active status: 1
1001@10.8.5.71	<sip:1001@10.8.5.71:5060>;expires=180; Active status: 1
1100@10.8.5.71	<sip:1100@10.8.5.71:5060>;expires=180; Active status: 1
1101@10.8.5.71	<sip:1101@10.8.5.71:5060>;expires=180; Active status: 1
2000@10.8.5.72	<sip:2000@10.8.5.72:5060>;expires=180; Active status: 1

Table 32-1: SAS/SBC Registered Users Parameters

Column Name	Description
Address of Record	An address-of-record (AOR) is a SIP or SIPS URI that points to a domain with a location service that can map the URI to another URI (Contact) where the user might be available.
Contact	SIP URI that can be used to contact that specific instance of the User Agent for subsequent requests.

32.4 Viewing Call Routing Status

The Call Routing Status page provides you with information on the current routing method used by the device. This information includes the IP address and FQDN (if used) of the Proxy server with which the device currently operates.

To view the call routing status:

Open the Call Routing Status page (**Status & Diagnostics** tab >**VoIP Status** menu >**Call Routing Status**).

Figure 32-4: Call Routing Status Page

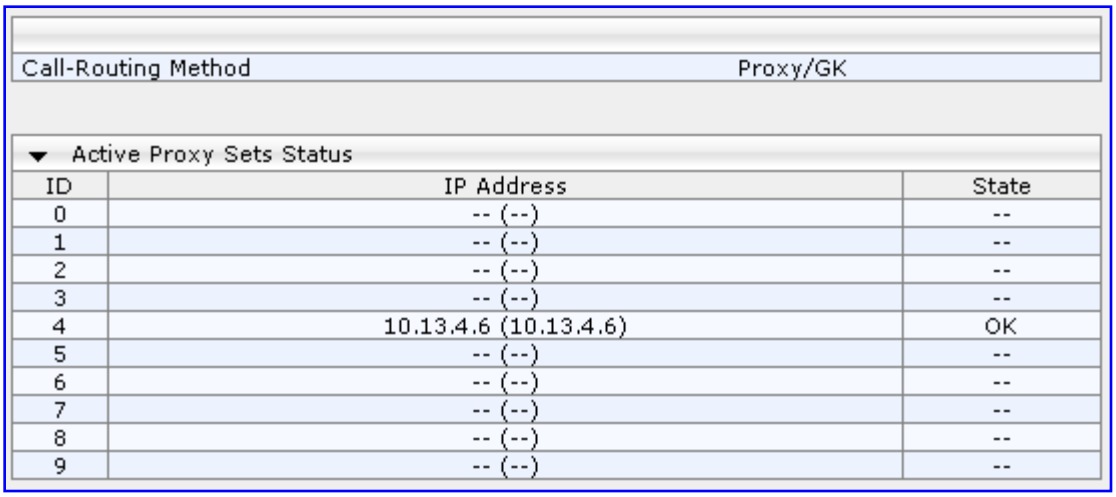


Table 32-2: Call Routing Status Parameters

Parameter	Description
Call-Routing Method	Proxy/GK = Proxy server is used to route calls.
IP Address	Not Used = Proxy server isn't defined. IP address and FQDN (if exists) of the Proxy server with which the device currently operates.
State	N/A = Proxy server isn't defined. OK = Communication with the Proxy server is in order. Fail = No response from any of the defined Proxies.

33 Reporting Information to External Party

33.1 Generating Call Detail Records

The Call Detail Record (CDR) contains vital statistic information on calls made from the device. CDRs are generated at the end and optionally, at the beginning of each call (defined by the CDRReportLevel parameter). Once generated, they are sent to a Syslog server. The destination IP address for CDR logs is defined by the CDRSyslogServerIP parameter. For CDR in RADIUS format, see 'Supported RADIUS Attributes' on page 274.

33.1.1 CDR Fields for SBC Signaling

The CDR fields for SBC signaling are listed in the table below. The signaling CDRs are published for each SBC leg.

Table 33-1: CDR Fields for SBC Signaling

CDR Field Name	Description
SBCReportType	Report Type (call start, connect, or end)
EPTyp	Endpoint type
SIPCallId	Unique ID of call
SessionId	Unique Session ID
Orig	Call originator ("LCL" for local; "RMT" for remote)
SourceIp	Source IP address
SourcePort	Source UDP port
DestIp	Destination IP address
DestPort	Destination UDP port
TransportType	Transport type (UDP, TCP, or TLS)
SrcURI	Source URI
SrcURIBeforeMap	Source URI before manipulation
DstURI	Destination URI
DstURIBeforeMap	Destination URI before manipulation
Durat	Call duration
TrmSd	Termination side (local or remote)
TrmReason	Termination reason
TrmReasonCategory	Termination reason category
SetupTime	Call setup time
ConnectTime	Call connect time
ReleaseTime	Call release time
RedirectReason	Redirect reason
RedirectURINum	Redirection URI
RedirectURINumBeforeMap	Redirect URI number before manipulation
TxSigIPDiffServ	Signaling IP DiffServ

CDR Field Name	Description
IPGroup	IP Group description
SrdId	SRD name
SIPInterfaceId	SIP Interface ID
ProxySetId	Proxy Set ID
IpProfileId	IP Profile ID
MediaRealm	Media Realm name
DirectMedia	Direct media or traversing SBC (yes or no)

33.1.2 CDR Fields for SBC Media

The CDR fields for SBC media are listed in the table below. The media CDRs are published for each active media stream, thereby allowing multiple media CDRs, where each media CDR has a unique call ID corresponding to the signaling CDR.

Table 33-2: CDR Fields for SBC Media

CDR Field Name	Description
MediaReportType	Report type (media start, update, or end)
SIPCallId	Unique call ID
Cid	Channel CID
MediaType	Media type (audio, video, or text)
Coder	Coder name
PacketInterval	Coder packet interval
LocalRtpIp	Local RTP IP address
LocalRtpPort	Local RTP port
RemoteRtpIp	Remote RTP IP address
RemoteRtpPort	Remote RTP port
InPackets	Number of received packets
OutPackets	Number of sent packets
LocalPackLoss	Local packet loss
RemotePackLoss	Remote packet loss
RTPdelay	RTP delay
RTPjitter	RTP jitter
TxRTPssrc	Tx RTP SSRC
RxRTPssrc	Local RTP SSRC
LocalRFactor	Local conversation quality
RemoteRFactor	Remote conversation quality
LocalMosCQ	Local MOS for conversation
RemoteMosCQ	Remote MOS for conversation

CDR Field Name	Description
TxRTPIPDiffServ	Media IP DiffServ

33.1.3 Supported RADIUS Attributes

The following table provides descriptions on the RADIUS attributes included in the communication packets transmitted between the device and a RADIUS Server.

Table 33-3: Supported RADIUS Attributes

Attribute Number	Attribute Name	VSA No.	Purpose	Value Format	Example	AAA[1]
Request Attributes						
1	User-Name		Account number or calling party number or blank	String up to 15 digits long	5421385747	Start Acc Stop Acc
4	NAS-IP-Address		IP address of the requesting device	Numeric	192.168.14.43	Start Acc Stop Acc
6	Service-Type		Type of service requested	Numeric	1: login	Start Acc Stop Acc
26	H323-Incoming-Conf-Id	1	SIP call identifier	Up to 32 octets		Start Acc Stop Acc
26	H323-Remote-Address	23	IP address of the remote gateway	Numeric		Stop Acc
26	H323-Conf-ID	24	H.323/SIP call identifier	Up to 32 octets		Start Acc Stop Acc
26	H323-Setup-Time	25	Setup time in NTP format 1	String		Start Acc Stop Acc
26	H323-Call-Origin	26	The call's originator: Answering (IP) or Originator (PSTN)	String	Answer, Originate etc	Start Acc Stop Acc
26	H323-Call-Type	27	Protocol type or family used on this leg of the call	String	VoIP	Start Acc Stop Acc
26	H323-Connect-Time	28	Connect time in NTP format	String		Stop Acc
26	H323-Disconnect-Time	29	Disconnect time in NTP format	String		Stop Acc
26	H323-Disconnect-Cause	30	Q.931 disconnect cause code	Numeric		Stop Acc
26	H323-Gw-ID	33	Name of the gateway	String	SIPIDString	Start Acc Stop Acc
26	SIP-Call-ID	34	SIP Call ID	String	abcde@ac.com	Start Acc Stop Acc

Attribute Number	Attribute Name	VSA No.	Purpose	Value Format	Example	AAA[1]
26	Call-Terminator	35	The call's terminator: PSTN-terminated call (Yes); IP-terminated call (No).	String	Yes, No	Stop Acc
30	Called-Station-ID			String	8004567145	Start Acc
			Destination phone number	String	2427456425	Stop Acc
			Calling Party Number (ANI)	String	5135672127	Start Acc Stop Acc
			Account Request Type (start or stop) **Note:** 'start' isn't supported on the Calling Card application.	Numeric	1: start, 2: stop	Start Acc Stop Acc
			No. of seconds tried in sending a particular record	Numeric	5	Start Acc Stop Acc
			Number of octets received for that call duration	Numeric		Stop Acc
			Number of octets sent for that call duration	Numeric		Stop Acc
			A unique accounting identifier - match start & stop	String	34832	Start Acc Stop Acc
			For how many seconds the user received the service	Numeric		Stop Acc
			Number of packets received during the call	Numeric		Stop Acc
			Number of packets sent during the call	Numeric		Stop Acc
			Physical port type of device on which the call is active	String	0: Asynchronous	Start Acc Stop Acc
Response Attributes						
26	H323-Return-Code	103	The reason for failing authentication (0 = ok, other number failed)	Numeric	0 Request accepted	Stop Acc
44	Acct-Session-ID		A unique accounting identifier – match start & stop	String		Stop Acc

Below is an example of RADIUS Accounting, where the non-standard parameters are preceded with brackets.

```
Accounting-Request (361)
user-name = 111
acct-session-id = 1
nas-ip-address = 212.179.22.213
nas-port-type = 0
acct-status-type = 2
acct-input-octets = 4841
acct-output-octets = 8800
acct-session-time = 1
acct-input-packets = 122
acct-output-packets = 220
called-station-id = 201
calling-station-id = 202
// Accounting non-standard parameters:
(4923 33) h323-gw-id =
(4923 23) h323-remote-address = 212.179.22.214
(4923 1) h323-ivr-out = h323-incoming-conf-id:02102944 600a1899
3fd61009 0e2f3cc5
(4923 30) h323-disconnect-cause = 22 (0x16)
(4923 27) h323-call-type = VOIP
(4923 26) h323-call-origin = Originate
(4923 24) h323-conf-id = 02102944 600a1899 3fd61009 0e2f3cc5
```

Part VIII

Diagnostics

This part describes the diagnostics procedures.

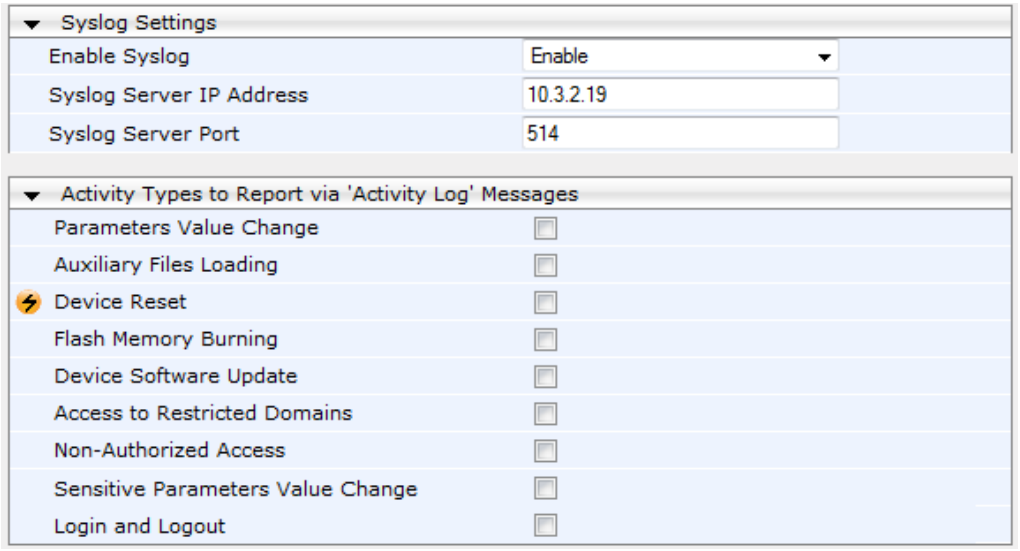
34 Configuring Syslog Settings

The Syslog Settings page allows you to configure the device's embedded Syslog client. For a detailed description on the Syslog parameters, see 'Syslog, CDR and Debug Parameters' on page 299. For more information on Syslog messages and using third-party Syslog servers, refer to the *Product Reference Manual*.

To configure the Syslog client:

1. Open the Syslog Settings page (**Configuration** tab >**System** menu >**Syslog Settings**).

Figure 34-1: Syslog Settings Page



Configure the parameters as required, and then click Submit to apply your changes.

To save the changes to flash memory, see 'Saving Configuration' on page 245.

Part IX

Appendices

This part includes appendices.

Configuration Parameters Reference

The device's configuration parameters, default values, and their descriptions are documented in this section.

Parameters and values enclosed in square brackets (**[...]) represent** the *ini* file parameters and their enumeration values; parameters not enclosed in square brackets represent their corresponding Web interface.

Note: Some parameters are configurable only through the *ini* file.

Networking Parameters

This subsection describes the device's networking parameters.

Ethernet Parameters

The Ethernet parameters are described in the table below.

Table A-1: Ethernet Parameters

Parameter	Description
Web: Physical Ports Settings [PhysicalPortsTable]	This parameter table configures the physical Ethernet ports. The format of this parameter is as follows: [PhysicalPortsTable] FORMAT PhysicalPortsTable_Index = PhysicalPortsTable_Port, PhysicalPortsTable_Mode, PhysicalPortsTable_NativeVlan, PhysicalPortsTable_SpeedDuplex, PhysicalPortsTable_PortDescription, PhysicalPortsTable_GroupMember, PhysicalPortsTable_GroupStatus; [\PhysicalPortsTable] For example: PhysicalPortsTable 0 = GE_4_1, 1, 1, 4, "User Port #0", GROUP_1, Active; PhysicalPortsTable 1 = GE_4_2, 1, 1, 4, "User Port #1", GROUP_1, Redundant;

VoIP Multiple Network Interfaces and VLAN Parameters

The IP network interfaces and VLAN parameters are described in the table below.

TableA-2: IP Network Interfaces and VLAN Parameters

Parameter	Description
Multiple Interface Table	
Web: Multiple Interface Table **EMS: IP Interface Settings** CLI: configure voip > interface network-if display **[InterfaceTable]**	This *para*meter table configures the Multiple Interface table for configuring the IP addresses of the voice and/or data functionalities and logical IP addresses. The format of this parameter is as follows: **[**InterfaceTable] FORMAT InterfaceTable_Index = InterfaceTable_ApplicationTypes, InterfaceTable_InterfaceMode, InterfaceTable_IPAddress, InterfaceTable_PrefixLength, InterfaceTable_Gateway, InterfaceTable_VlanID, InterfaceTable_InterfaceName, InterfaceTable_PrimaryDNSServerIPAddress, InterfaceTable_SecondaryDNSServerIPAddress, InterfaceTable_UnderlyingInterface; [\InterfaceTable] For example: InterfaceTable 0 = 0, 0, 192.168.85.14, 16, 0.0.0.0, 1, Management; InterfaceTable 1 = 2, 0, 200.200.85.14, 24, 0.0.0.0, 200, Control; InterfaceTable 2 = 1, 0, 211.211.85.14, 24, 211.211.85.1, 211, Media; **Notes:** For this *para*meter table to take effect, a device reset is required. Up to 48 logical IP addresses with associated VLANs can be defined (indices 0-47). Each interface index must be unique. Each interface must have a unique VLAN ID. Each interface must have a unique subnet. Subnets in different interfaces must not overlap (e.g., defining two interfaces with 10.0.0.1/8 and 10.50.10.1/24 is invalid). Each interface must have its own address space. Upon device start up, this table is parsed and passes comprehensive validation tests. If any errors occur during this validation phase, the device sends an error message to the Syslog server and falls back to a “safe mode”, using a single IPv4 interface and without VLANs. Therefore, check the Syslog for any error messages. To configure multiple VoIP IP interfaces in the Web interface and for a detailed description of the table's parameters, see 'Configuring IP Interface Settings' on page 68). For a description of configuring *ini* file table parameters, see 'Configuring ini File Table Parameters' on page 54.
[EnableNTPasOAM]	Defines the application type for NTP services. **[1]** = OAMP (default) **[0]** = Control. **Note:** For this parameter to take effect, a device reset is required.

VoIP Static Routing Parameters

The static routing parameters are described in the table below.

Table A-3: Static Routing Parameters

Parameter	Description
Static IP Routing Table	
Web/EMS: IP Routing Table CLI: configure voip > static **[StaticRouteTable]**	Defines up to 30 static VoIP IP routing rules for the device. These rules can be associated with IP interfaces defined in the Multiple Interface table (InterfaceTable parameter). The routing decision for sending the outgoing IP packet is based on the source subnet/VLAN. If not associated with an IP interface, the static IP rule is based on destination IP address. When the destination of an outgoing IP packet does not match one of the subnets defined in the Multiple Interface table, the device searches this table for an entry that matches the requested destination host/network. If such an entry is found, the device sends the packet to the indicated router (i.e., next hop). If no explicit entry is found, the packet is sent to the default gateway according to the source interface of the packet (if defined). The format of this parameter is as follows: [StaticRouteTable] FORMAT StaticRouteTable_Index = StaticRouteTable_InterfaceName, StaticRouteTable_Destination, StaticRouteTable_PrefixLength, StaticRouteTable_Gateway, StaticRouteTable_Description; [\StaticRouteTable] **Notes:** The Gateway address must be in the same subnet as configured in the Multiple Interface table for VoIP network interfaces (refer to 'Configuring IP Interface Settings' on page 68). The StaticRouteTable_Description parameter is a string value of up to 30 characters. The metric value (next hop) is automatically set to 1.

|Quality of Service Parameters

The Quality of Service (QoS) parameters are described in the table below.

The device allows you to specify DiffServ (Differentiated Services) values for four predifined service classes:

Premium Media service class – used for RTP Media traffic

Premium Control Service class – used for Call Control traffic

Gold Service class – used for streaming applications

Bronze Service class – used for OAMP applications

The Layer-3 QoS parameters enables setting the values of the DiffServ field in the IP Header of the frames related to a specific service class. The Layer-2 QoS parameters enable setting the values for the 3 priority bits in the VLAN tag (IEEE 802.1p standard) according to the value of the DiffServ field found in the packet IP header.

Table A-4: QoS Parameters

Parameter	Description
Layer-2 Class Of Service (CoS) Parameters (VLAN Tag Priority Field)	
Web: DiffServ Table EMS: QoS Settings – DSCP to QoS Mapping CLI: configure voip > vlan-mapping **[DiffServToVlanPriority]**	This parameter table allows you to configure DiffServ-to-VLAN Priority mapping. For each packet sent to the LAN, the VLAN Priority of the packet is set according to the DiffServ value in the IP header of the packet. The format of this ini file is as follows: [DiffServToVlanPriority] FORMAT DiffServToVlanPriority_Index = DiffServToVlanPriority_DiffServ, DiffServToVlanPriority_VlanPriority; [\DiffServToVlanPriority] For example: DiffServToVlanPriority 0 = 46, 6; DiffServToVlanPriority 1 = 40, 6; DiffServToVlanPriority 2 = 26, 4; DiffServToVlanPriority 3 = 10, 2; **Notes:** For this parameter to take effect, a device reset is required. You can configure up to 64 VLAN tag priorities (i.e., indices 0-63). The valid range of the parameter DiffServ is 0-63. The valid range of the parameter VlanPriority is 0-7. To set a default VLAN Priority (used for ARPs and automated ICMP packets, as well as for IP Packets without DiffServ value), set a VLAN Priority value for DiffServ 0.
Layer-3 Class of Service (TOS/DiffServ) Parameters	
Web: Media Premium QoS EMS: Premium Service Class Media Diff Serv CLI: media-qos **[PremiumServiceClassMediaDiffServ]**	Defines the DiffServ value for Premium Media CoS content. The valid range is 0 to 63. The default value is 46. **Notes:** For this parameter to take effect, a device reset is required. The value for the Premium Control DiffServ is determined by the following (according to priority): IPDiffServ value in the selected IP Profile (IPProfile parameter).

Parameter	Description
	PremiumServiceClassMediaDiffServ.
Web: Control Premium QoS EMS: Premium Service Class Control Diff Serv CLI: control-qos **[PremiumServiceClassControlDiffServ]**	Defines the DiffServ value for Premium Control CoS content (Call Control applications). The valid range is 0 to 63. The default value is 40. **Notes:** For this parameter to take effect, a device reset is required. The value for the Premium Control DiffServ is determined by the following (according to priority): SigIPDiffserv value in the selected IP Profile (IPProfile parameter). PremiumServiceClassControlDiffServ.
Web: Gold QoS EMS: Gold Service Class Diff Serv CLI: gold-qos **[GoldServiceClassDiffServ]**	Defines the DiffServ value for the Gold CoS content (Streaming applications). The valid range is 0 to 63. The default value is 26. **Note:** For this parameter to take effect, a device reset is required.
Web: Bronze QoS EMS: Bronze Service Class Diff Serv CLI: bronze-qos **[BronzeServiceClassDiffServ]**	Defines the DiffServ value for the Bronze CoS content (OAMP applications). The valid range is 0 to 63. The default value is 10. **Note:** For this parameter to take effect, a device reset is required.

NAT Parameters

The Network Address Translation (NAT) parameters are described in the table below.

Table A-5: NAT Parameters

Parameter	Description
NAT Parameters	
Web/EMS: NAT Traversal CLI: disable-NAT-traversal **[DisableNAT]**	Enables the NAT mechanism. **[0]** Enable **[1]** Disable (default) **Note:** The compare operation that is performed on the IP address is enabled by default and is configured by the parameter EnableIPAddrTranslation. The compare operation that is performed on the UDP port is disabled by default and is configured by the parameter EnableUDPPortTranslation.
Web: NAT IP Address EMS: Static NAT IP Address CLI: nat-ip-addr **[StaticNatIP]**	Defines the global (public) IP address of the device to enable static NAT between the device and the Internet. **Note:** For this parameter to take effect, a device reset is required.
[EnableIPAddrTranslation]	Enables IP address translation for RTP, RTCP, and T.38 packets. **[0]** = Disable IP address translation. **[1]** = Enable IP address translation (default). When enabled, the device compares the source IP address of the first incoming packet to the remote IP address stated in the opening of the channel. If the two IP addresses don't match, the NAT mechanism is activated. Consequently, the remote IP address of the outgoing stream is replaced by the source IP address of the first incoming packet. **Note:** The NAT mechanism must be enabled for this parameter to take effect (i.e., the parameter DisableNAT is set to 0).
[EnableUDPPortTranslation]	Enables UDP port translation. **[0]** = Disables UDP port translation (default). **[1]** = Enables UDP port translation. The device compares the source UDP port of the first incoming packet to the remote UDP port stated in the opening of the channel. If the two UDP ports don't match, the NAT mechanism is activated. Consequently, the remote UDP port of the outgoing stream is replaced by the source UDP port of the first incoming packet. **Notes:** For this parameter to take effect, a device reset is required. The NAT mechanism and the IP address translation must be enabled for this parameter to take effect (i.e., set the parameter DisableNAT to 0 and the parameter EnableIpAddrTranslation to 1).

NFS Parameters

The Network File Systems (NFS) configuration parameters are described in the table below.

Table A-6: NFS Parameters

Parameter	Description
CLI: base-port **[NFSBasePort]**	Defines the start of the range of numbers used for local UDP ports used by the NFS client. The maximum number of local ports is maximum channels plus maximum NFS servers. The valid range is 0 to 65535. The default is 47000.
Web: NFS Table **EMS: NFS Settings**	
[NFSServers]	This *para*meter table defines up to 16 NFS file systems so that the device can access a remote server's shared files and directories for loading cmp, ini, and auxiliary files (using the Automatic Update mechanism). As a file system, the NFS is independent of machine types, OSs, and network architectures. Note that an NFS file server can share multiple file systems. There must be a separate row for each remote file system shared by the NFS file server that needs to be accessed by the device. The format of this ini file table parameter is as follows: [NFSServers] FORMAT NFSServers_Index = NFSServers_HostOrIP, NFSServers_RootPath, NFSServers_NfsVersion, NFSServers_AuthType, NFSServers_UID, NFSServers_GID, NFSServers_VlanType; [\NFSServers] For example: NFSServers 1 = 101.1.13, /audio1, 3, 1, 0, 1, 1; **Notes:** You can configure up to 16 NFS file systems (where the first index is 0). To avoid terminating current calls, a row must not be deleted or modified while the device is currently accessing files on the remote NFS file system. The combination of host/IP and Root Path must be unique for each index in the table. For example, the table must include only one index entry with a Host/IP of '192.168.1.1' and Root Path of '/audio'. This parameter is applicable only if VLANs are enabled or Multiple IPs is configured. For a detailed description of the table's parameters and to configure NFS using the Web interface, see 'Configuring NFS Settings' on page 95. For a description of configuring *ini* file table parameters, see 'Configuring ini File Table Parameters' on page 54.

DNS Parameters

The Domain name System (DNS) parameters are described in the table below.

Table A-7: DNS Parameters

Parameter	Description
Internal DNS Table	
Web: Internal DNS Table EMS: DNS Information CLI: configure voip > control-network dns Dns2Ip **[DNS2IP]**	This *para*meter table defines the internal DNS table for resolving host names into IP addresses. Up to four different IP addresses (in dotted-decimal notation) can be assigned to a host name. The format of this parameter is as follows: [Dns2Ip] FORMAT Dns2Ip_Index = Dns2Ip_DomainName, Dns2Ip_FirstIpAddress, Dns2Ip_SecondIpAddress, Dns2Ip_ThirdIpAddress, Dns2Ip_FourthIpAddress; [\Dns2Ip] For example: Dns2Ip 0 = DnsName, 1.1.1.1, 2.2.2.2, 3.3.3.3, 4.4.4.4; **Notes:** This parameter can include up to 20 indices. If the internal DNS table is used, the device first attempts to resolve a domain name using this table. If the domain name isn't found, the device performs a DNS resolution using an external DNS server. To configure the internal DNS table using the Web interface and for a description of the parameters in this *ini* file table parameter, see 'Configuring the Internal DNS Table' on page 91. For configuring *ini* file table parameters, see 'Configuring ini File Table Parameters' on page 54.
Internal SRV Table	
Web: Internal SRV Table EMS: DNS Information CLI: configure voip > control-network dns Srv2Ip **[SRV2IP]**	This parameter table defines the internal SRV table for resolving host names into DNS A-Records. Three different A-Records can be assigned to a host name. Each A-Record contains the host name, priority, weight, and port. The format of this parameter is as follows: [SRV2IP] FORMAT SRV2IP_Index = SRV2IP_InternalDomain, SRV2IP_TransportType, SRV2IP_Dns1, SRV2IP_Priority1, SRV2IP_Weight1, SRV2IP_Port1, SRV2IP_Dns2, SRV2IP_Priority2, SRV2IP_Weight2, SRV2IP_Port2, SRV2IP_Dns3, SRV2IP_Priority3, SRV2IP_Weight3, SRV2IP_Port3; [\SRV2IP] For example: SRV2IP 0 = SrvDomain,0,Dnsname1,1,1,500,Dnsname2,2,2,501,$$,0,0,0; **Notes**: This parameter can include up to 10 indices. If the Internal SRV table is used, the device first attempts to resolve a domain name using this table. If the domain name isn't located, the device performs an SRV resolution using an external DNS server. To configure the Internal SRV table using the Web interface and for a description of the parameters in this *ini* file table parameter, see 'Configuring the Internal SRV Table' on page 92. For configuring *ini* file table parameters, see 'Configuring ini File Table

Parameter	Description
	Parameters' on page 54.

DHCP Parameters

The Dynamic Host Control Protocol (DHCP) parameters are described in the table below.

Table A-8: DHCP Parameters

Parameter	Description
Web: Enable DHCP EMS: DHCP Enable **[DHCPEnable]**	Enables Dynamic Host Control Protocol (DHCP) functionality. **[0]** Disable = Disable DHCP support on the device (default). **[1]** Enable = Enable DHCP support on the device. **Notes:** For this parameter to take effect, a device reset is required. After you enable the DHCP server, perform the following procedure: Enable DHCP and save the configuration. Perform a cold reset using the device's hardware reset button (soft reset using the Web interface doesn't trigger the DHCP procedure and this parameter reverts to 'Disable'). For more information on DHCP, refer to the *Product Reference Manual*. This parameter is a special 'Hidden' parameter. Once defined and saved in flash memory, its assigned value doesn't revert to its default even if the parameter doesn't appear in the *ini* file.
EMS: DHCP Speed Factor **[DHCPSpeedFactor]**	Defines the DHCP renewal speed. **[0]** = Disable **[1]** = Normal (default) **[2]** to **[10]** = Fast When set to 0, the DHCP lease renewal is disabled. Otherwise, the renewal time is divided by this factor. Some DHCP-enabled routers perform better when set to 4. **Note:** For this parameter to take effect, a device reset is required.

NTP and Daylight Saving Time Parameters

The Network Time Protocol (NTP) and daylight saving time parameters are described in the table below.

Table A-9: NTP and Daylight Saving Time Parameters

Parameter	Description
NTP Parameters **Note:** For more information on Network Time Protocol (NTP), see 'Simple Network Time Protocol Support' on page 65.	
Web: NTP Server IP Address EMS: Server IP Address CLI: primary-server **[NTPServerIP]**	Defines the IP address (in dotted-decimal notation) of the NTP server. The default IP address is 0.0.0.0 (i.e., internal NTP client is disabled).
Web: NTP UTC Offset EMS: UTC Offset CLI: utc-offset **[NTPServerUTCOffset]**	Defines the Universal Time Coordinate (UTC) offset (in seconds) from the NTP server. The default offset is 0. The offset range is -43200 to 43200.
Web: NTP Update Interval EMS: Update Interval CLI: update-interval **[NTPUpdateInterval]**	Defines the time interval (in seconds) that the NTP client requests for a time update. The default interval is 86400 (i.e., 24 hours). The range is 0 to 214783647. **Note:** It is not recommend to set this parameter to beyond one month (i.e., 2592000 seconds).
Daylight Saving Time Parameters	
Web: Day Light Saving Time EMS: Mode CLI: summer-time **[DayLightSavingTimeEnable]**	Enables daylight saving time. **[0]** Disable (default) **[1]** Enable
Web: Start Time EMS: Start CLI: start **[DayLightSavingTimeStart]**	Defines the date and time when daylight saving begins. The format of the value is mo:dd:hh:mm (month, day, hour, and minutes).
Web: End Time EMS: End CLI: end **[DayLightSavingTimeEnd]**	Defines the date and time when daylight saving ends. The format of the value is mo:dd:hh:mm (month, day, hour, and minutes).
Web/EMS: Offset CLI: offset **[DayLightSavingTimeOffset]**	Defines the daylight saving time offset (in minutes). The valid range is 0 to 120. The default is 60.

Management Parameters

This subsection describes the device's Web and Telnet parameters.

General Parameters

The general management parameters are described in the table below.

Table A-10: General Management Parameters

Parameter	Description
Web: Web and Telnet Access List Table EMS: Web Access Addresses **[WebAccessList_x]**	Defines up to ten IP addresses that are permitted to access the device's Web interface and Telnet interfaces. Access from an undefined IP address is denied. When no IP addresses are defined in this table, this security feature is inactive (i.e., the device can be accessed from any IP address). The default value is 0.0.0.0 (i.e., the device can be accessed from any IP address). For example: WebAccessList_0 = 10.13.2.66 WebAccessList_1 = 10.13.77.7 For defining the Web and Telnet Access list using the Web interface, see 'Configuring Web and Telnet Access List' on page 43.
Web: Use RADIUS for Web/Telnet Login EMS: Web Use Radius Login CLI: enable-mgmt-login **[WebRADIUSLogin]**	Enables RADIUS queries for Web and Telnet authentication. **[0]** Disable (default). **[1]** Enable = Logging into the device's Web and Telnet embedded servers is done through a RADIUS server. The device contacts a user-defined server and verifies the given user name and password against a remote database, in a secure manner. **Notes:** The parameter EnableRADIUS must be set to 1. RADIUS authentication requires HTTP basic authentication, meaning the user name and password are transmitted in clear text over the network. Therefore, it's recommended to set the parameter HTTPSOnly to 1 to force the use of HTTPS, since the transport is encrypted. If using RADIUS authentication when logging in to the CLI, only the primary Web User Account (which has Security Administration access level) can access the device's CLI (see 'Configuring Web User Accounts' on page 38).

Web Parameters

The Web parameters are described in the table below.

Table A-11: Web Parameters

Parameter	Description
Web: Deny Access On Fail Count **[DenyAccessOnFailCount]**	Defines the maximum number of login attempts after which the requesting IP address is blocked. The valid value range is 0 to 32768. The values 0 and 1 mean immediate block. The default is 3.
Web: Deny Authentication Timer **[DenyAuthenticationTimer]**	Defines the time (in seconds) that login to the Web interface is denied for a user that has reached maximum login attempts as defined by the DenyAccessOnFailCount parameter. Only after this time expires can the user attempt to login from the same IP address. The default is 0.
Web: Display Login Information **[DisplayLoginInformation]**	Enables display of user's login information on each successful login attempt. **[0]** = Disable (default) **[1]** = Enable
[EnableMgmtTwoFactorAuthentication]	Enables Web login authentication using a third-party, smart card. **[0]** = Disable (default) **[1]** = Enable When enabled, the device retrieves the Web user's login username from the smart card, which is automatically displayed (read-only) in the Web Login screen; the user is then required to provide only the login password. Typically, a TLS connection is established between the smart card and the device's Web interface, and a RADIUS server is implemented to authenticate the password with the username. Thus, this feature implements a two-factor authentication - what the user has (the physical card) and what the user knows (i.e., the login password).
[DisableWebTask]	Enables device management through the Web interface. **[0]** = Enable Web management (default). **[1]** = Disable Web management. **Note:** For this parameter to take effect, a device reset is required.
CLI: http-port **[HTTPport]**	Defines the LAN HTTP port for Web management (default is 80). To enable Web management from the LAN, configure the desired port. **Note:** For this parameter to take effect, a device reset is required.

Parameter	Description
EMS: Disable WEB Config **[DisableWebConfig]**	Determines whether the entire Web interface is read-only. **[0]** = Enables modifications of parameters (default). **[1]** = Web interface is read-only. When in read-only mode, parameters can't be modified. In addition, the following pages can't be accessed: 'Web User Accounts', 'Certificates', 'Regional Settings', 'Maintenance Actions' and all file-loading pages ('Load Auxiliary Files', 'Software Upgrade Wizard', and 'Configuration File'). **Note:** For this parameter to take effect, a device reset is required.
[ResetWebPassword]	Determines whether the device resets the username and password of the primary and secondary accounts to their default settings. **[0]** = Password and username retain their values (default). **[1]** = Password and username are reset. **Notes:** For this parameter to take effect, a device reset is required. The username and password cannot be reset from the Web interface (i.e., via AdminPage or by loading an *ini* file).
[WelcomeMessage]	This *para*meter table defines the Welcome message that appears after a Web interface login. The format of this parameter is as follows: [WelcomeMessage] FORMAT WelcomeMessage_Index = WelcomeMessage_Text [\WelcomeMessage] For Example: FORMAT WelcomeMessage_Index = WelcomeMessage_Text WelcomeMessage 1 = "**********" ; WelcomeMessage 2 = "********* This is a Welcome message ***" ; WelcomeMessage 3 = "**********" ; **Notes:** Each index represents a line of text in the Welcome message box. Up to 20 indices can be defined. The configured text message must be enclosed in double quotation marks (i.e., "..."). If this parameter is not configured, no Welcome message is displayed. For a description on using *ini* file table parameters, see 'Configuring ini File Table Parameters' on page 54.

Telnet Parameters

The Telnet parameters are described in the table below. **Note:** Telnet is currently supported only for debugging from the LAN interface.

Table A-12: Telnet Parameters

Parameter	Description
Web: Embedded Telnet Server EMS: Server Enable CLI: telnet **[TelnetServerEnable]**	Enables the device's embedded Telnet server. Telnet is disabled by default for security. **[0]** Disable (default) **[1]** Enable Unsecured **[2]** Enable Secured (SSL) **Note:** Only the primary Web User Account (which has Security Administration access level) can access the device using Telnet (see 'Configuring Web User Accounts' on page 38).
Web: Telnet Server TCP Port EMS: Server Port CLI: telnet-port **[TelnetServerPort]**	Defines the port number for the embedded Telnet server. The valid range is all valid port numbers. The default port is 23.
Web: Telnet Server Idle Timeout EMS: Server Idle Disconnect CLI: idle-timeout **[TelnetServerIdleDisconnect]**	Defines the timeout (in minutes) for disconnection of an idle Telnet session. When set to zero, idle sessions are not disconnected. The valid range is any value. The default value is 0. **Note:** For this parameter to take effect, a device reset is required.

SNMP Parameters

The SNMP parameters are described in the table below.

Table A-13: SNMP Parameters

Parameter	Description
Web: Enable SNMP CLI: disable **[DisableSNMP]**	Enables SNMP. **[0]** Enable = SNMP is enabled (default). **[1]** Disable = SNMP is disabled and no traps are sent.
CLI: port **[SNMPPort]**	Defines the device's local (LAN) UDP port used for SNMP Get/Set commands. The range is 100 to 3999. The default port is 161. **Note:** For this parameter to take effect, a device reset is required.
CLI: trusted-managers **[SNMPTrustedMGR_x]**	Defines up to five IP addresses of remote trusted SNMP managers from which the SNMP agent accepts and processes SNMP Get and Set requests. **Notes:** By default, the SNMP agent accepts SNMP Get and Set requests from any IP address, as long as the correct community string is used in the request. Security can be enhanced by using Trusted Managers, which is an IP address from which the SNMP agent accepts and processes SNMP requests. If no values are assigned to these parameters any manager can access the device. Trusted managers can work with all community strings.
[ChassisPhysicalAlias]	Defines the 'alias' name object for the physical entity as specified by a network manager, and provides a non-volatile 'handle' for the physical entity. The valid range is a string of up to 255 characters.
[ChassisPhysicalAssetID]	Defines the user-assigned asset tracking identifier object for the device's chassis as specified by an EMS, and provides non-volatile storage of this information. The valid range is a string of up to 255 characters.
[ifAlias]	Defines the textual name of the interface. The value is equal to the ifAlias SNMP MIB object. The valid range is a string of up to 64 characters.
EMS: Keep Alive Trap Port **[KeepAliveTrapPort]**	Defines the port to which keep-alive traps are sent. The valid range is 0 - 65534. The default is port 162.
[SendKeepAliveTrap]	Enables keep-alive traps and sends them every 9/10 of the time as defined by the NATBindingDefaultTimeout parameter. **[0]** = Disable **[1]** = Enable **Note:** For this parameter to take effect, a device reset is required.
CLI: sys-oid **[SNMPSysOid]**	Defines the base product system OID. The default is eSNMP_AC_PRODUCT_BASE_OID_D. **Note:** For this parameter to take effect, a device reset is required.

Parameter	Description
[SNMPTrapEnterpriseOid]	Defines the Trap Enterprise OID. The default is eSNMP_AC_ENTERPRISE_OID. The inner shift of the trap in the AcTrap subtree is added to the end of the OID in this parameter. **Note:** For this parameter to take effect, a device reset is required.
[acUserInputAlarmDescription]	Defines the description of the input alarm.
[acUserInputAlarmSeverity]	Defines the severity of the input alarm.
[AlarmHistoryTableMaxSize]	Defines the maximum number of rows in the Alarm History table. This parameter can be controlled by the Config Global Entry Limit MIB (located in the Notification Log MIB). The valid range is 50 to 1000. The default value is 500. **Note:** For this parameter to take effect, a device reset is required.
CLI: engine-id **[SNMPEngineIDString]**	Defines the SNMP engine ID for SNMPv2/SNMPv3 agents. This is used for authenticating a user attempting to access the SNMP agent on the device. The ID can be a string of up to 36 characters. The default value is 00:00:00:00:00:00:00:00:00:00:00:00 (12 Hex octets characters). The provided key must be set with 12 Hex values delimited by a colon (":") in the format xx:xx:...:xx. For example, 00:11:22:33:44:55:66:77:88:99:aa:bb **Notes:** For this parameter to take effect, a device reset is required. Before setting this parameter, all SNMPv3 users must be deleted; otherwise, the parameter setting is ignored. If the supplied key does not pass validation of the 12 Hex values input or it is set with the default value, the engine ID is generated according to RFC 3411.
Web: SNMP Trap Destination Parameters **EMS: Network > SNMP Managers Table** **Note:** Up to five SNMP trap managers can be defined.	
SNMP Manager CLI: is-used **[SNMPManagerIsUsed_x]**	Determines the validity of the parameters (IP address and port number) of the corresponding SNMP Manager used to receive SNMP traps. **[0]** (Check box cleared) = Disabled (default) **[1]** (Check box selected) = Enabled
Web: IP Address EMS: Address CLI: ip-address **[SNMPManagerTableIP_x]**	Defines the IP address of the remote host used as an SNMP Manager. The device sends SNMP traps to this IP address. Enter the IP address in dotted-decimal notation, e.g., 108.10.1.255.
Web: Trap Port EMS: Port CLI: port **[SNMPManagerTrapPort_x]**	Defines the port number of the remote SNMP Manager. The device sends SNMP traps to this port. The valid SNMP trap port range is 100 to 4000. The default port is 162.
Web: Trap Enable **[SNMPManagerTrapSendingEnable_x]**	Enables the sending of traps to the corresponding SNMP manager. **[0]** Disable = Sending is disabled. **[1]** Enable = Sending is enabled (default).

Parameter	Description
CLI: send-trap **[SNMPManagerTrapUser_x]**	This parameter can be set to the name of any configured SNMPV3 user to associate with this trap destination. This determines the trap format, authentication level, and encryption level. By default, the trap is associated with the SNMP trap community string.
Web: Trap Manager Host Name CLI: manager-host-name **[SNMPTrapManagerHostName]**	Defines an FQDN of a remote host that is used as an SNMP manager. The resolved IP address replaces the last entry in the Trap Manager table (defined by the parameter SNMPManagerTableIP_x) and the last trap manager entry of snmpTargetAddrTable in the snmpTargetMIB. For example: 'mngr.corp.mycompany.com'. The valid range is a 99-character string.
SNMP Community String Parameters	
Community String CLI: ro-community-string **[SNMPReadOnlyCommunityString_x]**	Defines up to five read-only SNMP community strings (up to 19 characters each). The default string is 'public'.
Community String CLI: rw-community-string **[SNMPReadWriteCommunityString_x]**	Defines up to five read/write SNMP community strings (up to 19 characters each). The default string is 'private'.
Trap Community String CLI: community-string **[SNMPTrapCommunityString]**	Defines the Community string used in traps (up to 19 characters). The default string is 'trapuser'.
Web: SNMP V3 Table **EMS: SNMP V3 Users**	
[SNMPUsers]	This *para*meter table defines SNMP v3 users. The format of this parameter is as follows: **[**SNMPUsers] FORMAT SNMPUsers_Index = SNMPUsers_Username, SNMPUsers_AuthProtocol, SNMPUsers_PrivProtocol, SNMPUsers_AuthKey, SNMPUsers_PrivKey, SNMPUsers_Group; [\SNMPUsers] For example: SNMPUsers 1 = v3admin1, 1, 0, myauthkey, -, 1; The example above configures user 'v3admin1' with security level authNoPriv(2), authentication protocol MD5, authentication text password 'myauthkey', and ReadWriteGroup2. **Notes:** This parameter can include up to 10 indices. For a description of this table's individual parameters and for configuring the table using the Web interface, see 'Configuring SNMP V3 Users' on page 51. For configuring *ini* file table parameters, see 'Configuring ini File Table Parameters' on page 54

Serial Parameters

The RS-232 serial parameters are described in the table below.

Table A-14: Serial Parameters

Parameter	Description
[DisableRS232]	Enables the device's RS-232 (serial) port. **[0]** = Enabled **[1]** = Disabled (default) The RS-232 serial port can be used to change the networking parameters and view error/notification messages. For how to establish a serial communication with the device, refer to the *Installation Manual*. **Note:** For this parameter to take effect, a device reset is required.
EMS: Baud Rate **[SerialBaudRate]**	Defines the RS-232 baud rate. The valid values include the following: 1200, 2400, 9600, 14400, 19200, 38400, 57600, or 115200 (default). **Note:** For this parameter to take effect, a device reset is required.
EMS: Data **[SerialData]**	Defines the RS-232 data bit. **[7]** = 7-bit. **[8]** = 8-bit (default). **Note:** For this parameter to take effect, a device reset is required.
EMS: Parity **[SerialParity]**	Defines the RS-232 polarity. **[0]** = None (default). **[1]** = Odd. **[2]** = Even. **Note:** For this parameter to take effect, a device reset is required.
EMS: Stop **[SerialStop]**	Defines the RS-232 stop bit. **[1]** = 1-bit (default). **[2]** = 2-bit. **Note:** For this parameter to take effect, a device reset is required.
EMS: Flow Control **[SerialFlowControl]**	Defines the RS-232 flow control. **[0]** = None (default). **[1]** = Hardware. **Note:** For this parameter to take effect, a device reset is required.

Debugging and Diagnostics Parameters

This subsection describes the device's debugging and diagnostic parameters.

General Parameters

The general debugging and diagnostic parameters are described in the table below.

Table A-15: General Debugging and Diagnostic Parameters

Parameter	Description
EMS: Enable Diagnostics **[EnableDiagnostics]**	Determines the method for verifying correct functioning of the different hardware components on the device. On completion of the check and if the test fails, the device sends information on the test results of each hardware component to the Syslog server. **[0]** = Rapid and Enhanced self-test mode (default). **[1]** = Detailed self-test mode (full test of DSPs, PCM, Switch, LAN, PHY and Flash). **[2]** = A quicker version of the Detailed self-test mode (full test of DSPs, PCM, Switch, LAN, PHY, but partial test of Flash). For more information, refer to the *Product Reference Manual*. **Note:** For this parameter to take effect, a device reset is required.
Web: Enable LAN Watchdog **[EnableLanWatchDog]**	Enables the LAN watchdog feature**.** **[0]** Disable (default). **[1]** Enable. When LAN watchdog is enabled, the device's overall communication integrity is checked periodically. If no communication is detected for about three minutes, the device performs a self test: If the self-test succeeds, the problem is a logical link down (i.e., Ethernet cable disconnected on the switch side) and the Busy Out mechanism is activated if enabled (i.e., the parameter EnableBusyOut is set to 1). If the self-test fails, the device restarts to overcome internal fatal communication error. **Notes:** For this parameter to take effect, a device reset is required. Enable LAN watchdog is relevant only if the Ethernet connection is full duplex.
Web: Delay After Reset **[sec]** CLI: delay-after-reset **[GWAppDelayTime]**	Defines the time interval (in seconds) that the device's operation is delayed after a reset. The valid range is 0 to 45. The default value is 7 seconds. **Note:** This feature helps overcome connection problems caused by some LAN routers or IP configuration parameters' modifications by a DHCP server.

Syslog, CDR and Debug Parameters

The Syslog, CDR and debug parameters are described in the table below.

Table A-16: Syslog, CDR and Debug Parameters

Parameter	Description
Web: Enable Syslog EMS: Syslog enable CLI: syslog **[EnableSyslog]**	Determines whether the device sends logs and error messages generated by the device to a Syslog server. **[0]** Disable= Logs and errors are not sent to the Syslog server (default). **[1]** Enable= Enables the Syslog server. **Notes:** If you enable Syslog, you must enter an IP address of the Syslog server (using the SyslogServerIP parameter). Syslog messages may increase the network traffic. To configure Syslog SIP message logging levels, use the GwDebugLevel parameter. For more information on Syslog, refer to the *Product Reference Manual*.
Web/EMS: Syslog Server IP Address CLI: syslog-ip **[SyslogServerIP]**	Defines the IP address (in dotted-decimal notation) of the computer on which the Syslog server is running. The Syslog server is an application designed to collect the logs and error messages generated by the device. Default IP address is 0.0.0.0. For information on Syslog, refer to the *Product Reference Manual*.
Web: Syslog Server Port EMS: Syslog Server Port Number CLI: syslog-port **[SyslogServerPort]**	Defines the UDP port of the Syslog server. The valid range is 0 to 65,535. The default port is 514. For information on Syslog, refer to the *Product Reference Manual*.
CLI: mx-syslog-lgth **[MaxBundleSyslogLength]**	Defines the maximum size (in bytes) threshold of logged Syslog messages bundled into a single UDP packet, after which they are sent to a Syslog server. The valid value range is 0 to 1220 (where 0 indicates that no bundling occurs). The default is 1220. **Note:** This parameter is applicable only if the GWDebugLevel parameter is set to 7.
Web: CDR Server IP Address EMS: IP Address of CDR Server CLI: cdr-srvr-ip-adrr **[CDRSyslogServerIP]**	Defines the destination IP address to where CDR logs are sent. The default value is a null string, which causes CDR messages to be sent with all Syslog messages to the Syslog server. **Notes:** The CDR messages are sent to UDP port 514 (default Syslog port). This mechanism is active only when Syslog is enabled (i.e., the parameter EnableSyslog is set to 1).
Web/EMS: CDR Report Level CLI: cdr-report-level **[CDRReportLevel]**	Determines whether Call Detail Records (CDR) are sent to the Syslog server and when they are sent. **[0]** None = CDRs are not used (default). **[1]** End Call = CDR is sent to the Syslog server at the end of each call. **[2]** Start & End Call = CDR report is sent to Syslog at the start and end of each call. **[3]** Connect & End Call = CDR report is sent to Syslog at connection and at the end of each call. **[4]** Start & End & Connect Call = CDR report is sent to Syslog at the start, at connection, and at the end of each call.

Parameter	Description
	Notes: The CDR Syslog message complies with RFC 3161 and is identified by: Facility = 17 (local1) and Severity = 6 (Informational). This mechanism is active only when Syslog is enabled (i.e., the parameter EnableSyslog is set to 1).
Web/EMS: Debug Level CLI: debug-level **[GwDebugLevel]**	Defines the Syslog debug logging level. **[0]** 0 (default) = Debug is disabled. **[1]** 1 = Flow debugging is enabled. **[5]** 5 = Flow, device interface, stack interface, session manager, and device interface expanded debugging are enabled. **[7]** 7 = This option is recommended when the device is running under "heavy" traffic. In this mode: The Syslog debug level automatically changes between level 5, level 1, and level 0, depending on the device's CPU consumption so that VoIP traffic isn't affected. Syslog messages are bundled into a single UDP packet, after which they are sent to a Syslog server (bundling size is determined by the MaxBundleSyslogLength parameter). Bundling reduces the number of UDP Syslog packets, thereby improving CPU utilization. Note that when this option is used, in order to read Syslog messages with Wireshark, a special plug-in (i.e., acsyslog.dll) must be used. Once the plug-in is installed, the Syslog messages are decoded as "AC SYSLOG" and are dispalyed using the 'acsyslog' filter instead of the regular 'syslog' filter. **Notes:** This parameter is typically set to 5 if debug traces are required. However, in cases of heavy traffic, option 7 is recommended. Options 2, 3, 4, and 6 are not recommended.
Syslog Facility Number **[SyslogFacility]**	Defines the Facility level (0 through 7) of the device's Syslog messages, according to RFC 3164. This allows you to identify Syslog messages generated by the device. This is useful, for example, if you collect the device's and other equipments' Syslog messages, at one single server. The device's Syslog messages can easily be identified and distinguished from other Syslog messages by its Facility level. Therefore, in addition to filtering Syslog messages according to IP address, the messages can be filtered according to Facility level. **[16]** = local use 0 (local0) - default **[17]** = local use 1 (local1) **[18]** = local use 2 (local2) **[19]** = local use 3 (local3) **[20]** = local use 4 (local4) **[21]** = local use 5 (local5) **[22]** = local use 6 (local6) **[23]** = local use 7 (local7)

Parameter	Description
Web: Activity Types to Report via Activity Log Messages **[ActivityListToLog]**	Defines the Activity Log mechanism of the device, which sends log messages (to a Syslog server) for reporting certain types of Web operations according to the below user-defined filters. **[pvc]** Parameters Value Change = Changes made on-the-fly to parameters. **[afl]** Auxiliary Files Loading = Loading of auxiliary files. **[dr]** Device Reset = Reset of device via the 'Maintenance Actions page. **Note:** For this option to take effect, a device reset is required. **[fb]** Flash Memory Burning = Burning of files or parameters to flash (in 'Maintenance Actions page). **[swu]** Device Software Update = cmp file loading via the Software Upgrade Wizard. **[ard]** Access to Restricted Domains = Access to restricted domains, which include the following Web pages: (1) ini parameters (AdminPage) (2) General Security Settings (3) Configuration File (5) Software Upgrade Key Status (7) Web & Telnet Access List (8) WEB User Accounts **[naa]** Non-Authorized Access = Attempt to access the Web interface with a false or empty user name or password. **[spc]** Sensitive Parameters Value Change = Changes made to sensitive parameters: (1) IP Address (2) Subnet Mask (3) Default Gateway IP Address (4) ActivityListToLog **[ll]** Login and Logout = Every login and logout attempt. For example: ActivityListToLog = 'pvc', 'afl', 'dr', 'fb', 'swu', 'ard', 'naa', 'spc' **Note:** For the *ini* file, values must be enclosed in single quotation marks.

Security Parameters

This subsection describes the device's security parameters.

General Parameters

The general security parameters are described in the table below.

Table A-17: General Security Parameters

Parameter	Description
Web: Internal Firewall Parameters **EMS: Firewall Settings** CLI: configure voip > access-list	
[AccessList]	This *para*meter table defines the device's access list (firewall), which defines network traffic filtering rules. For each packet received on the network interface, the table is scanned from the top down until a matching rule is found. This rule can either deny (block) or permit (allow) the packet. Once a rule in the table is located, subsequent rules further down the table are ignored. If the end of the table is reached without a match, the packet is accepted. The format of this parameter is as follows: [AccessList] FORMAT AccessList_Index = AccessList_Source_IP, AccessList_Source_Port, AccessList_PrefixLen, AccessList_Source_Port, AccessList_Start_Port, AccessList_End_Port, AccessList_Protocol, AccessList_Use_Specific_Interface, AccessList_Interface_ID, AccessList_Packet_Size, AccessList_Byte_Rate, AccessList_Byte_Burst, AccessList_Allow_Type; [\AccessList] For example: AccessList 10 = mgmt.customer.com, , , 32, 0, 80, tcp, 1, OAMP, 0, 0, 0, allow; AccessList 22 = 10.4.0.0, , , 16, 4000, 9000, any, 0, , 0, 0, 0, block; In the example above, Rule #10 allows traffic from the host 'mgmt.customer.com' destined to TCP ports 0 to 80 on interface OAMP (OAMP). Rule #22 blocks traffic from the subnet 10.4.xxx.yyy destined to ports 4000 to 9000. **Notes:** This parameter can include up to 50 indices. To configure the firewall using the Web interface and for a description of the parameters of this *ini* file table parameter, see 'Configuring Firewall Settings' on page 98. For a description of configuring with *ini* file table parameters, see 'Configuring ini File Table Parameters' on page 54.

HTTPS Parameters

The Secure Hypertext Transport Protocol (HTTPS) parameters are described in the table below.

Table A-18: HTTPS Parameters

Parameter	Description
Web: Secured Web Connection (HTTPS) EMS: HTTPS Only CLI: secured-connection **[HTTPSOnly]**	Determines the protocol used to access the Web interface. **[0]** HTTP and HTTPS (default). **[1]** HTTPs Only = Unencrypted HTTP packets are blocked. **Note:** For this parameter to take effect, a device reset is required.
EMS: HTTPS Port CLI: https-port **[HTTPSPort]**	Defines the local Secured HTTPS port of the device. This parameter allows secure remote device Web management from the LAN. To enable secure Web management from the LAN, configure the desired port. The valid range is 1 to 65535 (other restrictions may apply within this range). The default port is 443. **Note:** For this parameter to take effect, a device reset is required.
EMS: HTTPS Cipher String CLI: https-cipher-string **[HTTPSCipherString]**	Defines the Cipher string for HTTPS (in OpenSSL cipher list format). For the valid range values, refer to URL http://www.openssl.org/docs/apps/ciphers.html. The default value is 'EXP' (Export encryption algorithms). For example, use 'ALL' for all ciphers suites (e.g., for ARIA encryption for TLS). The only ciphers available are RC4 and DES, and the cipher bit strength is limited to 56 bits. **Notes:** If the "Strong Encryption" Software Upgrade Key is enabled, the default of the HTTPSCipherString parameter is changed to 'RC4:EXP', enabling RC-128bit encryption. The value 'ALL' can be configured only if the "Strong Encryption" Software Upgrade Key is enabled.
Web: HTTP Authentication Mode EMS: Web Authentication Mode CLI: http-auth-mode **[WebAuthMode]**	Determines the authentication mode used for the Web interface. **[0]** Basic Mode = Basic authentication (clear text) is used (default). **[1]** Digest When Possible = Digest authentication (MD5) is used. **[2]** Basic if HTTPS, Digest if HTTP = Digest authentication (MD5) is used for HTTP, and basic authentication is used for HTTPS. **Note:** When RADIUS login is enabled (i.e., the parameter WebRADIUSLogin is set to 1), basic authentication is forced.

Parameter	Description
CLI: req-client-cert **[HTTPSRequireClientCertificate]**	Determines whether client certificates are required for HTTPS connection. **[0]** = Client certificates are not required (default). **[1]** = Client certificates are required. The client certificate must be preloaded to the device and its matching private key must be installed on the managing PC. Time and date must be correctly set on the device for the client certificate to be verified. **Notes:** For this parameter to take effect, a device reset is required. For a description on implementing client certificates, see 'Client Certificates' on page 62.

SRTP Parameters

The Secure Real-Time Transport Protocol (SRTP) parameters are described in the table below.

Table A-19: SRTP Parameters

Parameter	Description
Web: Media Security EMS: Enable Media Security CLI: media-security-enable **[EnableMediaSecurity]**	Enables Secure Real-Time Transport Protocol (SRTP). **[0]** Disable = SRTP is disabled (default). **[1]** Enable = SRTP is enabled. **Note:** For this parameter to take effect, a device reset is required.
Web/EMS: Media Security Behavior **[MediaSecurityBehaviour]**	Determines the device's mode of operation when SRTP is used (i.e., when the parameter EnableMediaSecurity is set to 1). **[0]** Preferable = The device initiates encrypted calls. However, if negotiation of the cipher suite fails, an unencrypted call is established. Incoming calls that don't include encryption information are accepted. (default) **[1]** Mandatory = The device initiates encrypted calls, but if negotiation of the cipher suite fails, the call is terminated. Incoming calls that don't include encryption information are rejected. **[2]** Disable = The IP Profile for which this parameter is set does not support encrypted calls (i.e., SRTP). **[3]** Preferable - Single Media = The device sends SDP with a single media ('m=') line only (e.g., m=audio 6000 RTP/AVP 4 0 70 96) with RTP/AVP and crypto keys. The remote UA can respond with SRTP or RTP parameters: If the remote SIP UA does not support SRTP, it uses RTP and ignores the crypto lines. In the opposite direction, if the device receives an SDP offer with a single media (as shown above), it responds with SRTP (RTP/SAVP) if the EnableMediaSecurity parameter is set to 1. If SRTP is not supported (i.e., EnableMediaSecurity is set to 0), it responds with RTP. **Notes:** Before configuring this parameter, set the EnableMediaSecurity parameter to 1. If this parameter is set to Preferable [3] and two 'm=' lines are received in the SDP offer, the device prefers the SAVP (secure audio video profile) regardless of the order in the SDP. Option **[2]** Disable is applicable only to IP Profiles. This parameter can also be configured per IP Profile, using the IPProfile parameter (see 'Configuring IP Profiles' on page 137).
Web: Master Key Identifier (MKI) Size EMS: Packet MKI Size CLI: SRTP-tx-packet-MKI-size **[SRTPTxPacketMKISize]**	Defines the size (in bytes) of the Master Key Identifier (MKI) in SRTP Tx packets. The range is 0 to 4. The default value is 0.
CLI: symmetric-mki **[EnableSymmetricMKI]**	Enables symmetric MKI negotiation. **[0]** = Disabled (default) - the device includes the MKI in its 200 OK response according to the SRTPTxPacketMKISize parameter (if set to 0, then it is not included; if set to any other value, it is included with this value). **[1]** = Enabled - the answer crypto line contains (or excludes) an MKI

Parameter	Description

	value according to the selected crypto line in the offer. For example, assume that the device receives an INVITE containing the following two crypto lines in SDP:
	`a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:TAaxNnQt8/qLQMnDuG4vxYfWl6K7eBK/ufk04pR4|2^31 |1:1`
	`a=crypto:3 AES_CM_128_HMAC_SHA1_80 inline:bnuYZnMxSfUiGitviWJZmzr7OF3AiRO0l5Vnh0kH|2^31`
	The first crypto line includes the MKI parameter "1:1". In the 200 OK response, the device selects one of the crypto lines (i.e., '2' or '3'). If it selects crypto line '2', it includes the MKI parameter in its answer SDP, for example:
	`a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:R1VyA1xV/qwBjkEklu4kSJyl3wCtYeZLq1/QFuxw|2^31 |1:1`
	If the device selects a crypto line that does not contain the MKI parameter, then the MKI parameter is not included in the crypto line in the SDP answer (even if the SRTPTxPacketMKISize parameter is set to any value other than 0).
	Note: To enable symmetric MKI, the SRTPTxPacketMKISize parameter must be set to any value other than 0.

Web/EMS: SRTP offered Suites CLI: offer-srtp-cipher **[SRTPofferedSuites]**	Defines the offered crypto suites (cipher encryption algorithms) for SRTP. **[0]** = All available crypto suites (default) **[1]** CIPHER SUITES AES CM 128 HMAC SHA1 80 = device uses AES-CM encryption with a 128-bit key and HMAC-SHA1 message authentication with a 80-bit tag. **[2]** CIPHER SUITES AES CM 128 HMAC SHA1 32 = device uses AES-CM encryption with a 128-bit key and HMAC-SHA1 message authentication with a 32-bit tag. **Note:** This parameter also affects the selection of the crypto in the device's answer. For example, if the device receives an offer with two crypto lines containing HMAC_SHA1_80 and HMAC_SHA_32, it uses the HMAC_SHA_32 key in its SIP 200 OK response if the parameter is set to 2.
Web: Disable Authentication On Transmitted RTP Packets EMS: RTP AuthenticationDisable Tx CLI: RTP-authentication-disable-tx **[RTPAuthenticationDisableTx]**	Enables authentication on transmitted RTP packets in a secured RTP session. **[0]** Enable (default) **[1]** Disable
Web: Disable Encryption On Transmitted RTP Packets EMS: RTP EncryptionDisable Tx CLI: RTP-encryption-disable-tx **[RTPEncryptionDisableTx]**	Enables encryption on transmitted RTP packets in a secured RTP session. **[0]** Enable (default) **[1]** Disable
Web: Disable Encryption On Transmitted RTCP Packets EMS: RTCP EncryptionDisable Tx	Enables encryption on transmitted RTCP packets in a secured RTP session. **[0]** Enable (default) **[1]** Disable

Parameter	Description
CLI: RTCP-encryption-disable-tx **[RTCPEncryptionDisableTx]**	

TLS Parameters

The Transport Layer Security (TLS) parameters are described in the table below.

Table A-20: TLS Parameters

Parameter	Description
Web/EMS: TLS Version CLI: version **[TLSVersion]**	Determines the supported versions of SSL/TLS (Secure Socket Layer/Transport Layer Security. **[0]** SSL 2.0-3.0 and TLS 1.0 = SSL 2.0, SSL 3.0, and TLS 1.0 are supported (default). **[1]** TLS 1.0 Only = only TLS 1.0 is used. When set to 0, SSL/TLS handshakes always start with SSL 2.0 and switch to TLS 1.0 if both peers support it. When set to 1, TLS 1.0 is the only version supported; clients attempting to contact the device using SSL 2.0 are rejected. **Note:** For this parameter to take effect, a device reset is required.
Web: TLS Client Re-Handshake Interval EMS: TLS Re Handshake Interval CLI: tls-re-hndshk-int **[TLSReHandshakeInterval]**	Defines the time interval (in minutes) between TLS Re-Handshakes initiated by the device. The interval range is 0 to 1,500 minutes. The default is 0 (i.e., no TLS Re-Handshake).
Web: TLS Mutual Authentication EMS: SIPS Require Client Certificate **[SIPSRequireClientCertificate]**	Determines the device's behavior when acting as a server for TLS connections. **[0]** Disable = The device does not request the client certificate (default). **[1]** Enable = The device requires receipt and verification of the client certificate to establish the TLS connection. **Notes:** For this parameter to take effect, a device reset is required. The SIPS certificate files can be changed using the parameters HTTPSCertFileName and HTTPSRootFileName.
Web/EMS: Peer Host Name Verification Mode **[PeerHostNameVerificationMode]**	Determines whether the device verifies the Subject Name of a remote certificate when establishing TLS connections. **[0]** Disable = Disable (default). **[1]** Server Only = Verify Subject Name only when acting as a server for the TLS connection. **[2]** Server & Client = Verify Subject Name when acting as a server or client for the TLS connection. When a remote certificate is received and this parameter is not disabled, the value of SubjectAltName is compared with the list of available Proxies. If a match is found for any of the configured Proxies, the TLS connection is established. The comparison is performed if the SubjectAltName is either a DNS name (DNSName) or an IP address. If no match is found and the SubjectAltName is marked as 'critical', the TLS connection is not established. If DNSName is used, the certificate can also use wildcards ('*') to replace parts of the domain name. If the SubjectAltName is not marked as 'critical' and there is no match, the CN value of the SubjectName field is compared with the parameter TLSRemoteSubjectName. If a match is found, the connection is established. Otherwise, the connection is terminated.

Parameter	Description
Web: TLS Client Verify Server Certificate EMS: Verify Server Certificate CLI: tls-vrfy-srvr-cert **[VerifyServerCertificate]**	Determines whether the device, when acting as a client for TLS connections, verifies the Server certificate. The certificate is verified with the Root CA information. **[0]** Disable (default). **[1]** Enable. **Note:** If Subject Name verification is necessary, the parameter PeerHostNameVerificationMode must be used as well.
Web/EMS: TLS Remote Subject Name CLI: tls-rmt-subs-name **[TLSRemoteSubjectName]**	Defines the Subject Name that is compared with the name defined in the remote side certificate when establishing TLS connections. If the SubjectAltName of the received certificate is not equal to any of the defined Proxies Host names/IP addresses and is not marked as 'critical', the Common Name (CN) of the Subject field is compared with this value. If not equal, the TLS connection is not established. If the CN uses a domain name, the certificate can also use wildcards ('*') to replace parts of the domain name. The valid range is a string of up to 49 characters. **Note:** This parameter is applicable only if the parameter PeerHostNameVerificationMode is set to 1 or 2.
Web:Client Cipher String CLI: client-cipher-string **[TLSClientCipherString]**	Defines the cipher-suite string for TLS clients. The valid value is up to 255 strings. The default is "ALL:!ADH". For example: TLSClientCipherString = 'EXP' This parameter complements the HTTPSCipherString parameter (which affects TLS servers). For possible values and additional details, refer to: http://www.openssl.org/docs/apps/ciphers.html
CLI: pkey-size **[TLSPkeySize]**	Defines the key size (in bits) for RSA public-key encryption for newly self-signed generated keys for SSH. **[512]** **[768]** **[1024]** (default) **[2048]**

SSH Parameters

Secure Shell (SSH) parameters are described in the table below.

Table A-21: SSH Parameters

Parameter	Description
Web/EMS: SSH Server Enable CLI: ssh **[SSHServerEnable]**	Enables the device's embedded SSH server. **[0]** Disable (default) **[1]** Enable
Web/EMS: SSH Server Port cli: ssh-port **[SSHServerPort]**	Defines the port number for the embedded SSH server. Range is any valid port number. The default port is 22.
CLI: ssh-admin-key **[SSHAdminKey]**	Defines the RSA public key for strong authentication for logging in to the SSH interface (if enabled). The value should be a base64-encoded string. The value can be a maximum length of 511 characters. For more information, refer to the *Product Reference Manual*.
CLI: ssh-max-login-attempts **[SSHMaxLoginAttempts]**	Defines the maximum SSH login attempts allowed for entering an incorrect password by an administrator before the SSH session is rejected. The valid range is 1 to 3. the default is 3.
CLI: ssh-last-login-message **[SSHEnableLastLoginMessage]**	Enables message display in SSH sessions of the time and date of the last SSH login. The SSH login message displays the number of unsuccessful login attempts since the last successful login. **[0]** Disable **[1]** Enable (default) **Note:** The last SSH login information is cleared when the device is reset.
CLI: ssh-max-sessions **[SSHMaxSessions]**	Defines the maximum number of simultaneous SSH sessions. The valid range is 1 to 2. The default is 2 sessions.
CLI: ssh-require-public-key **[SSHRequirePublicKey]**	Enables RSA public keys for SSH. **[0]** = RSA public keys are optional if a value is configured for the parameter SSHAdminKey (default). **[1]** = RSA public keys are mandatory. **Note:** To define the key size, use the TLSPkeySize parameter.

OCSP Parameters

The Online Certificate Status Protocol (OCSP) parameters are described in the table below.

Table A-22: OCSP Parameters

Parameter	Description
Web: Enable OCSP Server EMS: OCSP Enable CLI: enable **[OCSPEnable]**	Enables or disables certificate checking using OCSP. **[0]** Disable (default). **[1]** Enable. For a description of OCSP, refer to the *Product Reference Manual*.
Web: Primary Server IP EMS: OCSP Server IP CLI: server-ip **[OCSPServerIP]**	Defines the IP address of the OCSP server. The default IP address is 0.0.0.0.
Web: Secondary Server IP CLI: secondary-server-ip **[OCSPSecondaryServerIP]**	Defines the IP address (in dotted-decimal notation) of the secondary OCSP server (optional). The default IP address is 0.0.0.0.
Web: Server Port EMS: OCSP Server Port CLI: server-port **[OCSPServerPort]**	Defines the OCSP server's TCP port number. The default port number is 2560.
Web: Default Response When Server Unreachable EMS: OCSP Default Response CLI: default-response **[OCSPDefaultResponse]**	Determines the default OCSP behavior when the server cannot be contacted. **[0]** Disable = Rejects peer certificate (default). **[1]** Enable = Allows peer certificate.

RADIUS Parameters

The RADIUS parameters are described in the table below. For supported RADIUS attributes, see 'Supported RADIUS Attributes' on page 274.

Table A-23: RADIUS Parameters

Parameter	Description
Web: Enable RADIUS Access Control CLI: enable **[EnableRADIUS]**	Enables the RADIUS application. **[0]** Disable = RADIUS application is disabled (default). **[1]** Enable = RADIUS application is enabled. **Note:** For this parameter to take effect, a device reset is required.
Web: Accounting Server IP Address CLI: accounting-server-ip **[RADIUSAccServerIP]**	Defines the IP address of the RADIUS accounting server.
Web: Accounting Port CLI: accounting-port **[RADIUSAccPort]**	Defines the port of the RADIUS accounting server. The default value is 1646.
Web/EMS: RADIUS Accounting Type CLI: radius-accounting **[RADIUSAccountingType]**	Determines when the RADIUS accounting messages are sent to the RADIUS accounting server. **[0]** At Call Release = Sent at call release only (default). **[1]** At Connect & Release = Sent at call connect and release. **[2]** At Setup & Release = Sent at call setup and release.
Web: AAA Indications EMS: Indications CLI: aaa-indications **[AAAIndications]**	Determines the Authentication, Authorization and Accounting (AAA) indications. **[0]** None = No indications (default). **[3]** Accounting Only = Only accounting indications are used.
Web: Device Behavior Upon RADIUS Timeout CLI: timeout-behavior **[BehaviorUponRadiusTimeout]**	Defines the device's response upon a RADIUS timeout. **[0]** Deny Access = Denies access. **[1]** Verify Access Locally = Checks password locally (default).
[MaxRADIUSSessions]	Defines the number of concurrent calls that can communicate with the RADIUS server (optional). The valid range is 0 to 240. The default value is 240.
[RADIUSRetransmission]	Defines the number of retransmission retries. The valid range is 1 to 10. The default value is 3.
[RadiusTO]	Defines the time interval (measured in seconds) that the device waits for a response before a RADIUS retransmission is issued. The valid range is 1 to 30. The default value is 10.
Web: RADIUS Authentication Server IP Address CLI: auth-server-ip **[RADIUSAuthServerIP]**	Defines the IP address of the RADIUS authentication server. **Note:** For this parameter to take effect, a device reset is required.
Web:RADIUS Authentication Server Port CLI: auth-server-port **[RADIUSAuthPort]**	Defines the port of the RADIUS Authentication Server. **Note:** For this parameter to take effect, a device reset is required.
Web: RADIUS Shared Secret	Defines the 'Secret' used to authenticate the device to the RADIUS

Parameter	Description
CLI: shared-secret **[SharedSecret]**	server. This should be a cryptically strong password.
Web: Default Access Level CLI: default-access-level **[DefaultAccessLevel]**	Defines the default access level for the device when the RADIUS (authentication) response doesn't include an access level attribute. The valid range is 0 to 255. The default value is 200 (Security Administrator').
Web: Local RADIUS Password Cache Mode CLI: local-cache-mode **[RadiusLocalCacheMode]**	Determines the device's mode of operation regarding the timer (configured by the parameter RadiusLocalCacheTimeout) that determines the validity of the user name and password (verified by the RADIUS server). **[0]** Absolute Expiry Timer = when you access a Web page, the timeout doesn't reset, instead it continues decreasing. **[1]** Reset Timer Upon Access = upon each access to a Web page, the timeout always resets (reverts to the initial value configured by RadiusLocalCacheTimeout).
Web: Local RADIUS Password Cache Timeout CLI: local-cache-timeout **[RadiusLocalCacheTimeout]**	Defines the time (in seconds) the locally stored user name and password (verified by the RADIUS server) are valid. When this time expires, the user name and password become invalid and a must be re-verified with the RADIUS server. The valid range is 1 to 0xFFFFFF. The default value is 300 (5 minutes). **[-1]** = Never expires. **[0]** = Each request requires RADIUS authentication.
Web: RADIUS VSA Vendor ID CLI: vsa-vendor-id **[RadiusVSAVendorID]**	Defines the vendor ID that the device accepts when parsing a RADIUS response packet. The valid range is 0 to 0xFFFFFFFF. The default value is 5003.
Web: RADIUS VSA Access Level Attribute CLI: vsa-access-level **[RadiusVSAAccessAttribute]**	Defines the code that indicates the access level attribute in the Vendor Specific Attributes (VSA) section of the received RADIUS packet. The valid range is 0 to 255. The default value is 35.

SIP Media Realm Parameters

The Media Realm parameters are described in the table below.

Table A-24: Media Realm Parameters

Parameter	Description
Media Realm Table	
Web: Media Realm Table **EMS: Protocol Definition > Media Realm** CLI: configure voip > media realm **[CpMediaRealm]**	This *para*meter table defines the Media Realm table. The Media Realm table allows you to divide a Media-type interface (defined in the Multiple Interface table) into several realms, where each realm is specified by a UDP port range. The format of this parameter is as follows: [CpMediaRealm] FORMAT CpMediaRealm_Index = CpMediaRealm_MediaRealmName, CpMediaRealm_IPv4IF, CpMediaRealm_IPv6IF, CpMediaRealm_PortRangeStart, CpMediaRealm_MediaSessionLeg, CpMediaRealm_PortRangeEnd, CpMediaRealm_TransRateRatio, CpMediaRealm_IsDefault; [\CpMediaRealm] For example, CpMediaRealm 1 = Mrealm1, Voice, , 6600, 20, 6790, , 1; CpMediaRealm 2 = Mrealm2, Voice, , 6800, 10, 6890; , 0; **Notes:** For this parameter to take effect, a device reset is required. This table can include up to 64 indices (where 0 is the first index). Each table index must be unique. A Media Realm can be assigned to an IP Group (in the IP Group table) or an SRD (in the SRD table). If different Media Realms are assigned to both an IP Group and SRD, the IP Group's Media Realm takes precedence. The parameter IPv6IF is not applicable. For a detailed description of all the parameters included in this *ini* file table parameter and for configuring Media Realms using the Web interface, see 'Configuring Media Realms' on page 105. For a description on configuring ini file table parameters, see 'Configuring ini File Table Parameters' on page 54.

Control Network Parameters

IP Group, Proxy, Registration and Authentication Parameters

The proxy server, registration and authentication SIP parameters are described in the table below.

Table A-25: Proxy, Registration and Authentication SIP Parameters

Parameter	Description
IP Group Table	
Web: IP Group Table EMS: Endpoints > IP Group CLI: configure voip > control-network ip-group **[IPGroup]**	This parameter table configures the IP Group table. The format of this parameter is as follows: [IPGroup] FORMAT IPGroup_Index = IPGroup_Type, IPGroup_Description, IPGroup_ProxySetId, IPGroup_SIPGroupName, IPGroup_ContactUser, IPGroup_EnableSurvivability, IPGroup_ServingIPGroup, IPGroup_SipReRoutingMode, IPGroup_AlwaysUseRouteTable, IPGroup_RoutingMode, IPGroup_SRD, IPGroup_MediaRealm, IPGroup_ClassifyByProxySet, IPGroup_ProfileId, IPGroup_MaxNumOfRegUsers, IPGroup_InboundManSet, IPGroup_OutboundManSet, IPGroup_RegistrationMode, IPGroup_AuthenticationMode, IPGroup_MethodList, IPGroup_EnableSBCClientForking, IPGroup_ContactName; [/IPGroup] For example: IPGroup 1 = 0, "dol gateway", 1, firstIPgroup, , 0, -1, 0, 0, -1, 0, mrealm1, 1, 1, ; IPGroup 2 = 0, "abc server", 2, secondIPgroup, , 0, -1, 0, 0, -1, 0, mrealm2, 1, 2, ; IPGroup 3 = 1, "IP phones", 1, thirdIPGroup, , 0, -1, 0, 0, -1, 0, mrealm3, 1, 2, ; **Notes:** For this parameter to take effect, a device reset is required. This table parameter can include up to 32 indices (where 1 is the first index). For a detailed description of the *ini* file table's parameters and for configuring this table using the Web interface, see 'Configuring IP Groups' on page 114. For configuring *ini* file table parameters, see 'Configuring ini File Table Parameters' on page 54.

Parameter	Description
Account Table	
Web: Account Table **EMS: SIP Endpoints > Account** CLI: configure voip > sip-definition account **[Account]**	This *para*meter table configures the Account table for registering and/or authenticating (digest) IP Groups (e.g., an IP-PBX) to a Serving IP Group (e.g., an Internet Telephony Service Provider - ITSP). The format of this parameter is as follows: [Account] FORMAT Account_Index = Account_ServedTrunkGroup, Account_ServedIPGroup, Account_ServingIPGroup, Account_Username, Account_Password, Account_HostName, Account_Register, Account_ContactUser, Account_ApplicationType; [\Account] For example: Account 1 = 1, -1, 1, user, 1234, acl, 1, ITSP1, 0; **Notes:** This table can include up to 32 indices (where 1 is the first index). For a detailed description of this table's parameters and for configuring this table using the Web interface, see 'Configuring Account Table' on page 132. For configuring *ini* file table parameters, see 'Configuring ini File Table Parameters' on page 54.
Proxy Registration Parameters	
Web: Use Default Proxy EMS: Proxy Used CLI: enable-proxy **[IsProxyUsed]**	Enables the use of a SIP proxy server. **[0]** No = Proxy isn't used and instead, the internal routing table is used (default). **[1]** Yes = Proxy server is used. Define the IP address of the proxy server in the Proxy Sets table (see 'Configuring Proxy Sets Table' on page 120).
Web/EMS: Proxy Name CLI: proxy-name **[ProxyName]**	Defines the Home Proxy domain name. If specified, this name is used as the Request-URI in REGISTER, INVITE, and other SIP messages, and as the host part of the To header in INVITE messages. If not specified, the Proxy IP address is used instead. The value must be string of up to 49 characters.
Web: Redundancy Mode EMS: Proxy Redundancy Mode CLI: redundancy-mode **[ProxyRedundancyMode]**	Determines whether the device switches back to the primary Proxy after using a redundant Proxy. **[0]** Parking = device continues working with a redundant (now active) Proxy until the next failure, after which it works with the next redundant Proxy (default). **[1]** Homing = device always tries to work with the primary Proxy server (i.e., switches back to the primary Proxy whenever it's available). **Note:** To use this Proxy Redundancy mechanism, you need to enable the keep-alive with Proxy option, by setting the parameter EnableProxyKeepAlive to 1 or 2.
Web: Proxy IP List Refresh Time EMS: IP List Refresh Time CLI: proxy-ip-lst-rfrsh-time **[ProxyIPListRefreshTime]**	Defines the time interval (in seconds) between each Proxy IP list refresh. The range is 5 to 2,000,000. The default interval is 60.
Web/EMS: Always Use Proxy CLI: always-use-proxy **[AlwaysSendToProxy]**	Determines whether the device sends SIP messages and responses through a Proxy server. **[0]** Disable = Use standard SIP routing rules (default).

Parameter	Description
	[1] Enable = All SIP messages and responses are sent to the Proxy server. **Note:** This parameter is applicable only if a Proxy server is used (i.e., the parameter IsProxyUsed is set to 1).
Web/EMS: DNS Query Type CLI: dns-query **[DNSQueryType]**	Enables the use of DNS Naming Authority Pointer (NAPTR) and Service Record (SRV) queries to resolve Proxy and Registrar servers and to resolve all domain names that appear in the SIP Contact and Record-Route headers. **[0]** A-Record (default) **[1]** SRV **[2]** NAPTR If set to A-Record **[0],** no NAPTR or SRV queries are performed. If set to SRV **[1]** and the Proxy/Registrar IP address parameter, Contact/Record-Route headers, or IP address defined in the Routing tables contain a domain name, an SRV query is performed. The device uses the first host name received from the SRV query. The device then performs a DNS A-record query for the host name to locate an IP address. If set to NAPTR **[2],** an NAPTR query is performed. If it is successful, an SRV query is sent according to the information received in the NAPTR response. If the NAPTR query fails, an SRV query is performed according to the configured transport type. If the Proxy/Registrar IP address parameter, the domain name in the Contact/Record-Route headers, or the IP address defined in the Routing tables contain a domain name with port definition, the device performs a regular DNS A-record query. If a specific Transport Type is defined, a NAPTR query is not performed. **Note:** To enable NAPTR/SRV queries for Proxy servers only, use the parameter ProxyDNSQueryType.
Web: Proxy DNS Query Type CLI: proxy-dns-query **[ProxyDNSQueryType]**	Enables the use of DNS Naming Authority Pointer (NAPTR) and Service Record (SRV) queries to discover Proxy servers. **[0]** A-Record (default) **[1]** SRV **[2]** NAPTR If set to A-Record **[0],** no NAPTR or SRV queries are performed. If set to SRV **[1]** and the Proxy IP address parameter contains a domain name without port definition (e.g., ProxyIP = domain.com), an SRV query is performed. The SRV query returns up to four Proxy host names and their weights. The device then performs DNS A-record queries for each Proxy host name (according to the received weights) to locate up to four Proxy IP addresses. Therefore, if the first SRV query returns two domain names and the A-record queries return two IP addresses each, no additional searches are performed. If set to NAPTR **[2],** an NAPTR query is performed. If it is successful, an SRV query is sent according to the information received in the NAPTR response. If the NAPTR query fails, an SRV query is performed according to the configured transport type. If the Proxy IP address parameter contains a domain name with port definition (e.g., ProxyIP = domain.com:5080), the device

Parameter	Description
	performs a regular DNS A-record query. If a specific Transport Type is defined, a NAPTR query is not performed. **Note:** When enabled, NAPTR/SRV queries are used to discover Proxy servers even if the parameter DNSQueryType is disabled.
Web/EMS: Password CLI: password-4-auth **[Password]**	Defines the password for Basic/Digest authentication with a Proxy/Registrar server. A single password is used for all device ports. The default is 'Default_Passwd'.
Web/EMS: Cnonce CLI: cnonce-4-auth **[Cnonce]**	Defines the Cnonce string used by the SIP server and client to provide mutual authentication. The value is free format, i.e., 'Cnonce = 0a4f113b'. The default is 'Default_Cnonce'.
Proxy IP Table	
Web:Proxy IP Table EMS: Proxy IP CLI: configure voip > control-network proxy-ip **[ProxyIP]**	This *para*meter table configures the Proxy Set table with Proxy Set IDs, each with up to five Proxy server IP addresses (or fully qualified domain name/FQDN). Each Proxy Set can be defined with a transport type (UDP, TCP, or TLS). The format of this parameter is as follows: [ProxyIP] FORMAT ProxyIp_Index = ProxyIp_IpAddress, ProxyIp_TransportType, ProxyIp_ProxySetId; [\ProxyIP] For example: ProxyIp 0 = 10.33.37.77, -1, 0; ProxyIp 1 = 10.8.8.10, 0, 2; ProxyIp 2 = 10.5.6.7, -1, 1; **Notes:** This parameter can include up to 32 indices (0-31). To assign various attributes (such as Proxy Load Balancing) per Proxy Set ID, use the parameter ProxySet. For configuring the Proxy Set ID table using the Web interface and for a detailed description of the parameters of this *ini* file table, see 'Configuring Proxy Sets Table' on page 120. For configuring *ini* file table parameters, see 'Configuring ini File Table Parameters' on page 54.

Parameter	Description
Proxy Set Table	
Web: Proxy Set Table EMS: Proxy Set CLI: configure voip > control-network proxy-set **[ProxySet]**	This *para*meter table configures the Proxy Set ID table. It is used in conjunction with the ProxyIP *ini* file table parameter, which defines the IP addresses per Proxy Set ID. The ProxySet *ini* file table parameter defines additional attributes per Proxy Set ID. This includes, for example, Proxy keep-alive and load balancing and redundancy mechanisms (if a Proxy Set contains more than one proxy address). The format of this parameter is as follows: [ProxySet] FORMAT ProxySet_Index = ProxySet_EnableProxyKeepAlive, ProxySet_ProxyKeepAliveTime, ProxySet_ProxyLoadBalancingMethod, ProxySet_IsProxyHotSwap, ProxySet_SRD, ProxySet_ClassificationInput, ProxySet_ProxyRedundancyMode; [\ProxySet] For example: ProxySet 0 = 0, 60, 0, 0, 0, , 1; ProxySet 1 = 1, 60, 1, 0, 1, , 0; **Notes:** This table parameter can include up to 32 indices (0-31). For configuring the Proxy Set IDs and their IP addresses, use the parameter ProxyIP. For configuring the Proxy Set ID table using the Web interface and for a detailed description of the parameters of this *ini* file table, see 'Configuring Proxy Sets Table' on page 120. For configuring *ini* file table parameters, see 'Configuring ini File Table Parameters' on page 54.
Registrar Parameters	
Web/EMS: Registration Time CLI: registration-time **[RegistrationTime]**	Defines the time interval (in seconds) for registering to a Proxy server. The value is used in the SIP Expires header. In addition, this parameter defines the time interval between Keep-Alive messages when the parameter EnableProxyKeepAlive is set to 2 (REGISTER). Typically, the device registers every 3,600 sec (i.e., one hour). The device resumes registration according to the parameter RegistrationTimeDivider. The valid range is 10 to 2,000,000. The default value is 180.
Web: Re-registration Timing **[%]** EMS: Time Divider CLI: re-registration-timing **[RegistrationTimeDivider]**	Defines the re-registration timing (in percentage). The timing is a percentage of the re-register timing set by the Registrar server. The valid range is 50 to 100. The default value is 50. For example: If this parameter is set to 70% and the Registration Expires time is 3600, the device re-sends its registration request after 3600 x 70% (i.e., 2520 sec). **Note**: This parameter may be overridden if the parameter RegistrationTimeThreshold is greater than 0.
Web/EMS: Registration Retry Time CLI: registration-retry-time **[RegistrationRetryTime]**	Defines the time interval (in seconds) after which a registration request is re-sent if registration fails with a 4xx response or if there is no response from the Proxy/Registrar server. The default is 30 seconds. The range is 10 to 3600.

Parameter	Description
Web: Registration Time Threshold EMS: Time Threshold CLI: registration-time-thres **[RegistrationTimeThreshold]**	Defines a threshold (in seconds) for re-registration timing. If this parameter is greater than 0, but lower than the computed re-registration timing (according to the parameter RegistrationTimeDivider), the re-registration timing is set to the following: timing set by the Registration server in the SIP Expires header minus the value of the parameter RegistrationTimeThreshold. The valid range is 0 to 2,000,000. The default value is 0.
Web: Re-register On INVITE Failure EMS: Register On Invite Failure CLI: reg-on-invite-fail **[RegisterOnInviteFailure]**	Enables immediate re-registration if no response is received for an INVITE request sent by the device. **[0]** Disable (default) **[1]** Enable When enabled, the device immediately expires its re-registration timer and commences re-registration to the same Proxy upon any of the following scenarios: The response to an INVITE request is 407 (Proxy Authentication Required) without an authentication header included. The remote SIP UA abandons a call before the device has received any provisional response (indicative of an outbound proxy server failure). The remote SIP UA abandons a call and the only provisional response the device has received for the call is 100 Trying (indicative of a home proxy server failure, i.e., the failure of a proxy in the route after the outbound proxy). The device terminates a call due to the expiration of RFC 3261 Timer B or due to the receipt of a 408 (Request Timeout) response and the device has not received any provisional response for the call (indicative of an outbound proxy server failure). The device terminates a call due to the receipt of a 408 (Request Timeout) response and the only provisional response the device has received for the call is the 100 Trying provisional response (indicative of a home proxy server failure).
Web: ReRegister On Connection Failure EMS: Re Register On Connection Failure CLI: reg-on-conn-failure **[ReRegisterOnConnectionFailure]**	Enables the device to perform SIP re-registration upon TCP/TLS connection failure. **[0]** Disable (default) **[1]** Enable
CLI: expl-un-reg **[UnregistrationMode]**	Enables the device to perform explicit unregisters. **[0]** Disable (default) **[1]** Enable = The device sends an asterisk ("*") value in the SIP Contact header, instructing the Registrar server to remove all previous registration bindings. The device removes SIP User Agent (UA) registration bindings in a Registrar, according to RFC 3261. Registrations are soft state and expire unless refreshed, but they can also be explicitly removed. A client can attempt to influence the expiration interval selected by the Registrar. A UA requests the immediate removal of a binding by specifying an expiration interval of "0" for that contact address in a REGISTER request. UA's should support this mechanism so that bindings can be removed before their expiration interval has passed. Use of the "*" Contact header field value allows a registering UA to remove all bindings

Parameter	Description
	associated with an address-of-record (AOR) without knowing their precise values. **Note:** The REGISTER-specific Contact header field value of "*" applies to all registrations, but it can only be used if the Expires header field is present with a value of "0".
Web/EMS: Add Empty Authorization Header CLI: add-empty-author-hdr **[EmptyAuthorizationHeader]**	Enables the inclusion of the SIP Authorization header in initial registration (REGISTER) requests sent by the device. **[0]** Disable (default) **[1]** Enable The Authorization header carries the credentials of a user agent (UA) in a request to a server. The sent REGISTER message populates the Authorization header with the following parameters: username - set to the value of the private user identity realm - set to the domain name of the home network uri - set to the SIP URI of the domain name of the home network nonce - set to an empty value response - set to an empty value For example: `Authorization: Digest username=alice_private@home1.net, realm=”home1.net”, nonce=””, response=”e56131d19580cd833064787ecc”` **Note:** This registration header is according to the IMS 3GPP TS24.229 and PKT-SP-24.220 specifications.

Web: Add initial Route Header
CLI: add-init-rte-hdr
[InitialRouteHeader]
	Enables the inclusion of the SIP Route header in initial registration or re-registration (REGISTER) requests sent by the device.
	[0] Disable (default)
	[1] Enable
	When the device sends a REGISTER message, the Route header includes either the Proxy's FQDN, or IP address and port according to the configured Proxy Set, for example:
	`Route: <sip:10.10.10.10;lr;transport=udp>`
	or
	`Route: <sip: pcscf-gm.ims.rr.com;lr;transport=udp>`

Parameter	Description
[UsePingPongKeepAlive]	Enables the use of the carriage-return and line-feed sequences (CRLF) Keep-Alive mechanism, according to RFC 5626 “Managing Client-Initiated Connections in the Session Initiation Protocol (SIP)” for reliable, connection-orientated transport types such as TCP. **[0]** Disable (default) **[1]** Enable The SIP user agent/client (i.e., device) uses a simple periodic message as a keep-alive mechanism to keep their flow to the proxy or registrar alive (used for example, to keep NAT bindings open). For connection-oriented transports such as TCP/TLS this is based on CRLF. This mechanism uses a client-to-server "ping" keep-alive and a corresponding server-to-client "pong" message. This ping-pong sequence allows the client, and optionally the server, to tell if its flow is still active and useful for SIP traffic. If the client does not receive a pong in response to its ping, it declares the flow “dead” and opens a new flow in its place. In the CRLF Keep-Alive mechanism the client periodically (defined by the PingPongKeepAliveTime parameter) sends a double-CRLF (the "ping") then waits to receive a single CRLF (the "pong"). If the client does not receive a "pong" within an appropriate amount of time, it considers the flow failed. **Note:** The device sends a CRLF message to the Proxy Set only if the Proxy Keep-Alive feature (EnableProxyKeepAlive parameter) is enabled and its transport type is set to TCP or TLS. The device first sends a SIP OPTION message to establish the TCP/TLS connection and if it receives any SIP response, it continues sending the CRLF keep-alive sequences.
[PingPongKeepAliveTime]	Defines the periodic interval (in seconds) after which a “ping” (double-CRLF) keep-alive is sent to a proxy/registrar, using the CRLF Keep-Alive mechanism. The default range is 5 to 2,000,000. The default is 120. The device uses the range of 80-100% of this user-defined value as the actual interval. For example, if the parameter value is set to 200 sec, the interval used is any random time between 160 to 200 seconds. This prevents an “avalanche” of keep-alive by multiple SIP UAs to a specific server.

Network Application Parameters

The SIP network application parameters are described in the table below.

Table A-26: SIP Network Application Parameters

Parameter	Description
Signaling Routing Domain Table	
Web: SRD Settings EMS: SRD Table CLI: config voip > control-network srd **[SRD]**	This *para*meter table configures the Signaling Routing Domain (SRD) table. The format of this parameter is as follows: [SRD] FORMAT SRD_Index = SRD_Name, SRD_MediaRealm, SRD_IntraSRDMediaAnchoring, SRD_BlockUnRegUsers, SRD_MaxNumOfRegUsers, SRD_EnableUnAuthenticatedRegistrations; [\SRD] For example: SRD 1 = LAN1_SRD, Mrealm1, 0, 1, 15, 1; SRD 2 = LAN2_SRD, Mrealm2, 0, 1, 15, 1; **Notes:** This table can include up to 32 indices (where 0 is the first index). For a detailed description of the table's individual parameters and for configuring the table using the Web interface, see 'Configuring SRD Table' on page 110. For a description on configuring *ini* file table parameters, see 'Configuring ini File Table Parameters' on page 54.
SIP Interface Table	
Web: SIP Interface Table EMS: SIP Interfaces Table CLI: configure voip > control-network sip-interface **[SIPInterface]**	This *para*meter table configures the SIP Interface table. The SIP Interface represents a SIP signaling entity, comprising ports (UDP, TCP, and TLS) and associated with a specific IP interface and an SRD ID. The format of this parameter is as follows: [SIPInterface] FORMAT SIPInterface_Index = SIPInterface_NetworkInterface, SIPInterface_ApplicationType, SIPInterface_UDPPort, SIPInterface_TCPPort, SIPInterface_TLSPort, SIPInterface_SRD; [\SIPInterface] For example: SIPInterface 0 = Voice, 2, 5060, 5060, 5061, 1; SIPInterface 1 = Voice, 2, 5070, 5070, 5071, 2; SIPInterface 2 = Voice, 0, 5090, 5000, 5081, 2; **Notes:** This table can include up to 32 indices (where 0 is the first index). Each SIP Interface must have a unique signaling port (i.e., no two SIP Interfaces can share the same port - no port overlapping). You can define up to three different SIP Interfaces per SRD, where each SIP Interface pertains to a different application type (i.e., GW, SAS, and SBC). For a detailed description of the table's individual parameters and for configuring the table using the Web interface, see 'Configuring SIP Interface Table' on page 112. For a description on configuring *ini* file table parameters, see 'Format of ini File Table Parameters' on page 54.

Parameter	Description
NAT Translation Table	
Web: NAT Translation Table CLI: configure voip > control-network NATTranslation **[NATTranslation]**	This *para*meter table defines NAT rules for translating source IP addresses per VoIP interface (SIP control and RTP media traffic) into NAT IP addresses. This allows, for example, the separation of VoIP traffic between different ISTP's, and topology hiding (of internal IP addresses to the "public" network). Each IP interface (configured in the Multiple Interface table - InterfaceTable parameter) can be associated with a NAT rule in this table, translating the source IP address and port of the outgoing packet into the NAT address (IP address and port range). The format of this parameter is as follows: [NATTranslation] FORMAT NATTranslation_Index = NATTranslation_SourceIPInterfaceName, NATTranslation_TargetIPAddress, NATTranslation_SourceStartPort, NATTranslation_SourceEndPort, NATTranslation_TargetStartPort, NATTranslation_TargetEndPort; [\NATTranslation] Where: SourceIPInterfaceName = name of the IP interface as defined in the Multiple Interface table. TargetIPAddress = global IP address. TargetStartPort and TargetEndPort = (optional) port range (1-65536) of the global address. If no ports are required, leave this field blank. SourceStartPort and SourceEndPort = (optional) port range (1-65536) of the IP interface. If no ports are required, leave this field blank. **Notes:** This table can include up to 32 indices. If NAT is not configured (by any of the above-mentioned methods), the device sends the packet according to its IP address defined in the Multiple Interface table.

General SIP Parameters

The general SIP parameters are described in the table below.

Table A-27: General SIP Parameters

Parameter	Description
Web/EMS: Max SIP Message Length **[KB]** **[MaxSIPMessageLength]**	Defines the maximum size (in Kbytes) for each SIP message that can be sent over the network. The device rejects messages exceeding this user-defined size. The valid value range is 1 to 50. The default is 50.
[SIPForceRport]	Determines whether the device sends SIP responses to the UDP port from where SIP requests are received even if the 'rport' parameter is not present in the SIP Via header. **[0]** (default) = Disabled - the device sends the SIP response to the UDP port defined in the Via header. If the Via header contains the 'rport' parameter, the response is sent to the UDP port from where the SIP request is received. **[1]** = Enabled - SIP responses are sent to the UDP port from where SIP requests are received even if the 'rport' parameter is not present in the Via header.
Web/EMS: PRACK Mode CLI: prack-mode **[PrackMode]**	Determines the PRACK (Provisional Acknowledgment) mechanism mode for SIP 1xx reliable responses. **[0]** Disable **[1]** Supported (default) **[2]** Required **Notes:** The Supported and Required headers contain the '100rel' tag. The device sends PRACK messages if 180/183 responses are received with '100rel' in the Supported or Required headers.
Web: Session-Expires Time EMS: Sip Session Expires CLI: session-expires-time **[SIPSessionExpires]**	Defines the numerical value sent in the Session-Expires header in the first INVITE request or response (if the call is answered). The valid range is 1 to 86,400 sec. The default is 0 (i.e., the Session-Expires header is disabled).
Web: Minimum Session-Expires EMS: Minimal Session Refresh Value CLI: min-session-expires **[MinSE]**	Defines the time (in seconds) that is used in the Min-SE header. This header defines the minimum time that the user agent refreshes the session. The valid range is 10 to 100,000. The default value is 90.
Web/EMS: Session Expires Method CLI: session-exp-method **[SessionExpiresMethod]**	Determines the SIP method used for session-timer updates. **[0]** Re-INVITE = Uses Re-INVITE messages for session-timer updates (default). **[1]** UPDATE = Uses UPDATE messages. **Notes:** The device can receive session-timer refreshes using both methods. The UPDATE message used for session-timer is excluded from the SDP body.

Parameter	Description
[RemoveToTagInFailureResponse]	Determines whether the device removes the 'to' header tag from final SIP failure responses to INVITE transactions. **[0]** = Do not remove tag (default). **[1]** = Remove tag.
[EnableRTCPAttribute]	Enables the use of the 'rtcp' attribute in the outgoing SDP. **[0]** = Disable (default) **[1]** = Enable
EMS: Options User Part **[OPTIONSUserPart]**	Defines the user part value of the Request-URI for outgoing SIP OPTIONS requests. A special value is 'empty', indicating that no user part in the Request-URI (host part only) is used. The valid range is a 30-character string. The default value is an empty string ('').
Web: SIP Transport Type EMS: Transport Type CLI: app-sip-transport-type **[SIPTransportType]**	Determines the default transport layer for outgoing SIP calls initiated by the device. **[0]** UDP (default) **[1]** TCP **[2]** TLS (SIPS) **Notes:** It's recommended to use TLS for communication with a SIP Proxy and not for direct device-to-device communication. For received calls (i.e., incoming), the device accepts all these protocols. The value of this parameter is also used by the SAS application as the default transport layer for outgoing SIP calls.
Web: SIP UDP Local Port EMS: Local SIP Port CLI: sip-udp-local-port **[LocalSIPPort]**	Defines the local UDP port for SIP messages. The valid range is 1 to 65534. The default value is 5060.
Web: SIP TCP Local Port EMS: TCP Local SIP Port CLI: sip-tcp-local-port **[TCPLocalSIPPort]**	Defines the local TCP port for SIP messages. The valid range is 1 to 65535. The default value is 5060.
Web: SIP TLS Local Port EMS: TLS Local SIP Port CLI: sip-tls-local-port **[TLSLocalSIPPort]**	Defines the local TLS port for SIP messages. The valid range is 1 to 65535. The default value is 5061. **Note:** The value of this parameter must be different from the value of the parameter TCPLocalSIPPort.
Web/EMS: Enable SIPS CLI: enable-sips **[EnableSIPS]**	Enables secured SIP (SIPS URI) connections over multiple hops. **[0]** Disable (default). **[1]** Enable. When the SIPTransportType parameter is set to 2 (i.e., TLS) and the parameter EnableSIPS is disabled, TLS is used for the next network hop only. When the parameter SIPTransportType is set to 2 or 1 (i.e., TCP or TLS) and EnableSIPS is enabled, TLS is used through the entire connection (over multiple hops). **Note:** If this parameter is enabled and the parameter SIPTransportType is set to 0 (i.e., UDP), the connection fails.
Web/EMS: Enable TCP Connection Reuse	Enables the reuse of the same TCP connection for all calls to the same destination.

Parameter	Description
CLI: tcp-conn-reuse **[EnableTCPConnectionReuse]**	**[0]** Disable = Use a separate TCP connection for each call. **[1]** Enable = Use the same TCP connection for all calls (default).
Web/EMS: Reliable Connection Persistent Mode CLI: reliable-conn-persistent **[ReliableConnectionPersistentMode]**	Enables setting of all TCP/TLS connections as persistent and therefore, not released. **[0]** = Disable (default) - all TCP connections (except those that are set to a proxy IP) are released if not used by any SIP dialog\transaction. **[1]** = Enable - TCP connections to all destinations are persistent and not released unless the device reaches 70% of its maximum TCP resources. While trying to send a SIP message connection, reuse policy determines whether live connections to the specific destination are re-used. Persistent TCP connection ensures less network traffic due to fewer setting up and tearing down of TCP connections and reduced latency on subsequent requests due to avoidance of initial TCP handshake. For TLS, persistent connection may reduce the number of costly TLS handshakes to establish security associations, in addition to the initial TCP connection set up. **Note:** If the destination is a Proxy server, the TCP/TLS connection is persistent regardless of the settings of this parameter.
Web/EMS: TCP Timeout CLI: tcp-timeout **[SIPTCPTimeout]**	Defines the Timer B (INVITE transaction timeout timer) and Timer F (non-INVITE transaction timeout timer), as defined in RFC 3261, when the SIP Transport Type is TCP. The valid range is 0 to 40 sec. The default value is 64*SIPT1Rtx msec.
Web: SIP Destination Port EMS: Destination Port CLI: sip-dst-port **[SIPDestinationPort]**	Defines the SIP destination port for sending initial SIP requests. The valid range is 1 to 65534. The default port is 5060. **Note:** SIP responses are sent to the port specified in the Via header.
Web: Tel to IP No Answer Timeout EMS: IP Alert Timeout CLI: tel2ip-no-ans-timeout **[IPAlertTimeout]**	Defines the time (in seconds) that the device waits for a 200 OK response from the called party (IP side) after sending an INVITE message. If the timer expires, the call is released. The valid range is 0 to 3600. The default value is 180.

Web/EMS: Enable GRUU
CLI: enable-gruu
[EnableGRUU]
	Determines whether the Globally Routable User Agent URIs (GRUU) mechanism is used, according to RFC 5627. This is used for obtaining a GRUU from a registrar and for communicating a GRUU to a peer within a dialog.
	[0] Disable (default)
	[1] Enable
	A GRUU is a SIP URI that routes to an instance-specific UA and can be reachable from anywhere. There are a number of contexts in which it is desirable to have an identifier that addresses a single UA (using GRUU) rather than the group of UA's indicated by an Address of Record (AOR). For example, in call transfer where user A is talking to user B, and user A wants to transfer the call to user C. User A sends a REFER to user C:
	`REFER sip:C@domain.com SIP/2.0`
	`From: sip:A@domain.com;tag=99asd`
	`To: sip:C@domain.com`

Parameter	Description
	`Refer-To: (URI that identifies B's UA)` The Refer-To header needs to contain a URI that user C can use to place a call to user B. This call needs to route to the specific UA instance that user B is using to talk to user A. User B should provide user A with a URI that has to be usable by anyone. It needs to be a GRUU. Obtaining a GRUU: The mechanism for obtaining a GRUU is through registrations. A UA can obtain a GRUU by generating a REGISTER request containing a Supported header field with the value “gruu”. The UA includes a “+sip.instance” Contact header parameter of each contact for which the GRUU is desired. This Contact parameter contains a globally unique ID that identifies the UA instance. The global unique ID is created from one of the following: If the REGISTER is per the device’s client (endpoint), it is the MAC address concatenated with the phone number of the client. If the REGISTER is per device, it is the MAC address only. When using TP, “User Info” can be used for registering per endpoint. Thus, each endpoint can get a unique id – its phone number. The globally unique ID in TP is the MAC address concatenated with the phone number of the endpoint. If the remote server doesn’t support GRUU, it ignores the parameters of the GRUU. Otherwise, if the remote side also supports GRUU, the REGISTER responses contain the “gruu” parameter in each Contact header. This parameter contains a SIP or SIPS URI that represents a GRUU corresponding to the UA instance that registered the contact. The server provides the same GRUU for the same AOR and instance-id when sending REGISTER again after registration expiration. RFC 5627 specifies that the remote target is a GRUU target if its’ Contact URL has the "gr" parameter with or without a value. Using GRUU: The UA can place the GRUU in any header field that can contain a URI. It must use the GRUU in the following messages: INVITE request, its 2xx response, SUBSCRIBE request, its 2xx response, NOTIFY request, REFER request and its 2xx response.

Web: User-Agent Information
EMS: User Agent Display Info
CLI: user-agent-info
[UserAgentDisplayInfo]
	Defines the string that is used in the SIP User-Agent and Server response headers. When configured, the string <UserAgentDisplayInfo value>/software version' is used, for example:
	`User-Agent: myproduct/v.6.00.010.006`
	If not configured, the default string, <Alcatel-Lucent product-name>/software version' is used, for example:
	`User-Agent: Alcatel-Lucent-Sip-Gateway-Mediant 1000 MSBG/v.6.00.010.006`
	The maximum string length is 50 characters.
	Note: The software version number and preceding forward slash (/) cannot be modified. Therefore, it is recommended not to include a forward slash in the parameter's value (to avoid two forward slashes in the SIP header, which may cause problems).

Web/EMS: SDP Session Owner CLI: sdp-session-owner	Defines the value of the Owner line ('o' field) in outgoing SDP messages.

Parameter	Description
[SIPSDPSessionOwner]	The valid range is a string of up to 39 characters. The default value is 'Alcatel-LucentGW'. For example: `o=Alcatel-LucentGW 1145023829 1145023705 IN IP4 10.33.4.126`
CLI: sdp-ver-nego **[EnableSDPVersionNegotiation]**	Enables the device to ignore new SDP re-offers (from the media negotiation perspective) in certain scenarios (such as session expires). According to RFC 3264, once an SDP session is established, a new SDP offer is considered a new offer only when the SDP origin value is incremented. In scenarios such as session expires, SDP negotiation is irrelevant and thus, the origin field is not changed. Even though some SIP devices don't follow this behavior and don't increment the origin value even in scenarios where they want to re-negotiate, the device can assume that the remote party operates according to RFC 3264, and in cases where the origin field is not incremented, the device does not re-negotiate SDP capabilities. **[0]** Disable = The device negotiates any new SDP re-offer, regardless of the origin field (default). **[1]** Enable = The device negotiates only an SDP re-offer with an incremented origin field.
Web/EMS: Subject CLI: usr-def-subject **[SIPSubject]**	Defines the Subject header value in outgoing INVITE messages. If not specified, the Subject header isn't included (default). The maximum length is up to 50 characters.
Web: Multiple Packetization Time Format EMS: Multi Ptime Format CLI: mult-ptime-format **[MultiPtimeFormat]**	Determines whether the 'mptime' attribute is included in the outgoing SDP. **[0]** None = Disabled (default) **[1]** PacketCable = includes the 'mptime' attribute in the outgoing SDP - PacketCable-defined format The mptime' attribute enables the device to define a separate Packetization period for each negotiated coder in the SDP. The mptime' attribute is only included if this parameter is enabled, even if the remote side includes it in the SDP offer. Upon receipt, each coder receives its 'ptime' value in the following precedence: from 'mptime' attribute, from 'ptime' attribute, and then from default value.
EMS: Enable P Time **[EnablePtime]**	Determines whether the 'ptime' attribute is included in the SDP. **[0]** = Remove the 'ptime' attribute from SDP. **[1]** = Include the 'ptime' attribute in SDP (default).
Web/EMS: 3xx Behavior CLI: 3xx-behavior **[3xxBehavior]**	Determines the device's behavior regarding call identifiers when a 3xx response is received for an outgoing INVITE request. The device can either use the same call identifiers (Call-ID, Branch, To, and From tags) or change them in the new initiated INVITE. **[0]** Forward = Use different call identifiers for a redirected INVITE message (default). **[1]** Redirect = Use the same call identifiers.
Web/EMS: Enable P-Charging Vector CLI: p-charging-vector **[EnablePChargingVector]**	Enables the inclusion of the P-Charging-Vector header to all outgoing INVITE messages. **[0]** Disable (default) **[1]** Enable

Parameter	Description
Web/EMS: Retry-After Time CLI: retry-aftr-time **[RetryAfterTime]**	Defines the time (in seconds) used in the Retry-After header when a 503 (Service Unavailable) response is generated by the device. The time range is 0 to 3,600. The default value is 0.
Web/EMS: Fake Retry After **[sec]** CLI: fake-retry-after **[FakeRetryAfter]**	Determines whether the device, upon receipt of a SIP 503 response without a Retry-After header, behaves as if the 503 response included a Retry-After header and with the period (in seconds) specified by this parameter. **[0]** Disable Any positive value (in seconds) for defining the period When enabled, this feature allows the device to operate with Proxy servers that do not include the Retry-After SIP header in SIP 503 (Service Unavailable) responses to indicate an unavailable service. The Retry-After header is used with the 503 (Service Unavailable) response to indicate how long the service is expected to be unavailable to the requesting SIP client. The device maintains a list of available proxies, by using the Keep-Alive mechanism. The device checks the availability of proxies by sending SIP OPTIONS every keep-alive timeout to all proxies. If the device receives a SIP 503 response to an INVITE, it also marks that the proxy is out of service for the defined "Retry-After" period.
Web/EMS: Enable P-Associated-URI Header CLI: p-associated-uri-hdr **[EnablePAssociatedURIHeader]**	Determines the device usage of the P-Associated-URI header. This header can be received in 200 OK responses to REGISTER requests. When enabled, the first URI in the P-Associated-URI header is used in subsequent requests as the From/P-Asserted-Identity headers value. **[0]** Disable (default). **[1]** Enable. **Note:** P-Associated-URIs in registration responses is handled only if the device is registered per endpoint (using the User Information file).
Web/EMS: Source Number Preference CLI: src-nb-preference **[SourceNumberPreference]**	Determines from which SIP header the source (calling) number is obtained in incoming INVITE messages. If not configured (i.e., empty string) or if any string other than "From" or "Pai2" is configured, the calling number is obtained from a specific header using the following logic: **a.** P-Preferred-Identity header. If the above header is not present, then the first P-Asserted-Identity header is used. If the above header is not present, then the Remote-Party-ID header is used. If the above header is not present, then the From header is used. "**From**" = The calling number is obtained from the From header. "**Pai2**" = The calling number is obtained using the following logic: **a.** If a P-Preferred-Identity header is present, the number is obtained from it. If no P-Preferred-Identity header is present and two P-Asserted-Identity headers are present, the number is obtained from the second P-Asserted-Identity header. If only one P-Asserted-Identity header is present, the calling

Parameter	Description
	number is obtained from it. **Notes:** The "From" and "Pai2" values are not case-sensitive. Once a URL is selected, all the calling party parameters are set from this header. If P-Asserted-Identity is selected and the Privacy header is set to 'id', the calling number is assumed restricted.
Web: Forking Timeout CLI: forking-timeout **[ForkingTimeOut]**	Defines the timeout (in seconds) that is started after the first SIP 2xx response has been received for a User Agent when a Proxy server performs call forking (Proxy server forwards the INVITE to multiple SIP User Agents). The device sends a SIP ACK and BYE in response to any additional SIP 2xx received from the Proxy within this timeout. Once this timeout elapses, the device ignores any subsequent SIP 2xx. The number of supported forking calls per channel is 20. In other words, for an INVITE message, the device can receive up to 20 forking responses from the Proxy server. The valid range is 0 to 30. The default is 30.
Web/EMS: Enable Reason Header CLI: reason-header **[EnableReasonHeader]**	Enables the usage of the SIP Reason header. **[0]** Disable **[1]** Enable (default)
Web/EMS: Gateway Name CLI: gw-name **[SIPGatewayName]**	Defines a name for the device (e.g., device123.com'). **Notes:** Ensure that the name defined is the one with which the Proxy is configured to identify the device. If specified, the device name is used as the host part of the SIP URI in the From header. If not specified, the device's IP address is used instead (default).
[ZeroSDPHandling]	Determines the device's response to an incoming SDP that includes an IP address of 0.0.0.0 in the SDP's Connection Information field (i.e., "c=IN IP4 0.0.0.0"). **[0]** = Sets the IP address of the outgoing SDP's c= field to 0.0.0.0 (default). **[1]** = Sets the IP address of the outgoing SDP c= field to the IP address of the device. If the incoming SDP doesn't contain the "a=inactive" line, the returned SDP contains the "a=recvonly" line.
Web/EMS: Enable Delayed Offer CLI: delayed-offer **[EnableDelayedOffer]**	Determines whether the device sends the initial INVITE message with or without an SDP. Sending the first INVITE without SDP is typically done by clients for obtaining the far-end's full list of capabilities before sending their own offer. (An alternative method for obtaining the list of supported capabilities is by using SIP OPTIONS, which is not supported by every SIP agent.) **[0]** Disable = The device sends the initial INVITE message with an SDP (default). **[1]** Enable = The device sends the initial INVITE message without an SDP.
[DisableCryptoLifeTimeInSDP]	Enables the device to send "a=crypto" lines without the lifetime parameter in the SDP. For example, if the SDP contains "a=crypto:12 AES_CM_128_HMAC_SHA1_80 inline:hhQe10yZRcRcpIFPkH5xYY9R1de37ogh9G1MpvNp|2^31

Parameter	Description
	", it removes the lifetime parameter "2^31". **[0]** Disable (default) **[1]** Enable
Web/EMS: Enable Contact Restriction CLI: contact-restriction **[EnableContactRestriction]**	Determines whether the device sets the Contact header of outgoing INVITE requests to 'anonymous' for restricted calls. **[0]** Disable (default) **[1]** Enable
EMS: P Asserted User Name **[PAssertedUserName]**	Defines a 'representative number' (up to 50 characters) that is used as the user part of the Request-URI in the P-Asserted-Identity header of an outgoing INVITE (for Tel-to-IP calls). The default value is null.
EMS: Use URL In Refer To Header **[UseAORInReferToHeader]**	Defines the source for the SIP URI set in the Refer-To header of outgoing REFER messages. **[0]** = Use SIP URI from Contact header of the initial call (default). **[1]** = Use SIP URI from To/From header of the initial call.
Web: Enable User-Information Usage CLI: user-inf-usage **[EnableUserInfoUsage]**	Enables the usage of the User Information, which is loaded to the device in the User Information auxiliary file. (For a description on User Information, see 'Loading Auxiliary Files' on page 247.) **[0]** Disable (default). **[1]** Enable
[HandleReasonHeader]	Determines whether the device uses the value of the incoming SIP Reason header for Release Reason mapping. **[0]** Disregard Reason header in incoming SIP messages. **[1]** Use the Reason header value for Release Reason mapping (default).
[EnableRport]	Enables the usage of the 'rport' parameter in the Via header. **[0]** = Disabled (default). **[1]** = Enabled. The device adds an 'rport' parameter to the Via header of each outgoing SIP message. The first Proxy that receives this message sets the 'rport' value of the response to the actual port from where the request was received. This method is used, for example, to enable the device to identify its port mapping outside a NAT. If the Via header doesn't include the 'rport' parameter, the destination port of the response is obtained from the host part of the Via header. If the Via header includes the 'rport' parameter without a port value, the destination port of the response is the source port of the incoming request. If the Via header includes 'rport' with a port value (e.g., rport=1001), the destination port of the response is the port indicated in the 'rport' parmeter.
[EnableRekeyAfter181]	Enables the device to send a Re-INVITE with a new (different) SRTP key (in the SDP) upon receipt of a SIP 181 response ("call is being forwarded"). **[0]** = Disable (default) **[1]** = Enable **Note:** This parameter is applicable only if SRTP is used.

Parameter	Description
[NumberOfActiveDialogs]	Defines the maximum number of active SIP dialogs that are not call related (i.e., REGISTER and SUBSCRIBE). This parameter is used to control the Registration/Subscription rate. The valid range is 1 to 20. The default value is 20.
Web: Enable Microsoft Extension CLI: microsoft-ext **[EnableMicrosoftExt]**	Enables the modification of the called and calling number for numbers received with Microsoft's proprietary "ext=xxx" parameter in the SIP INVITE URI user part. Microsoft Office Communications Server sometimes uses this proprietary parameter to indicate the extension number of the called or calling party. **[0]** Disable (default) **[1]** Enable For example, if a calling party makes a call to telephone number 622125519100 Ext. 104, the device receives the SIP INVITE (from Microsoft's application) with the URI user part as INVITE sip:622125519100;ext=104@10.1.1.10 (or INVITE tel:622125519100;ext=104). If the parameter EnableMicrosofExt is enabled, the device modifies the called number by adding an "e" as the prefix, removing the "ext=" parameter, and adding the extension number as the suffix (e.g., **e**622125519100**104**). Once modified, the device can then manipulate the number further, using the Number Manipulation tables to leave only the last 3 digits (for example) for sending to a PBX.
[TimeoutBetween100And18x]	Defines the timeout (in msec) between receiving a 100 Trying response and a subsequent 18x response. If a 18x response is not received within this timeout period, the call is disconnected. The valid range is 0 to 180,000 (i.e., 3 minutes). The default value is 32000 (i.e., 32 sec).
[IgnoreRemoteSDPMKI]	Determines whether the device ignores the Master Key Identifier (MKI) if present in the SDP received from the remote side. **[0]** Disable (default) **[1]** Enable
Retransmission Parameters	
Web: SIP T1 Retransmission Timer **[msec]** EMS: T1 RTX CLI: t1-re-tx-time **[SipT1Rtx]**	Defines the time interval (in msec) between the first transmission of a SIP message and the first retransmission of the same message. The default is 500. **Note:** The time interval between subsequent retransmissions of the same SIP message starts with SipT1Rtx. For INVITE requests, it is multiplied by two for each new retransmitted message. For all other SIP messages, it is multiplied by two until SipT2Rtx. For example, assuming SipT1Rtx = 500 and SipT2Rtx = 4000: The first retransmission is sent after 500 msec. The second retransmission is sent after 1000 (2*500) msec. The third retransmission is sent after 2000 (2*1000) msec. The fourth retransmission and subsequent retransmissions until SIPMaxRtx are sent after 4000 (2*2000) msec.

Parameter	Description
Web: SIP T2 Retransmission Timer **[msec]** EMS: T2 RTX CLI: t2-re-tx-time **[SipT2Rtx]**	Defines the maximum interval (in msec) between retransmissions of SIP messages (except for INVITE requests). The default is 4000. **Note:** The time interval between subsequent retransmissions of the same SIP message starts with SipT1Rtx and is multiplied by two until SipT2Rtx.
Web: SIP Maximum RTX EMS: Max RTX CLI: sip-max-rtx **[SIPMaxRtx]**	Defines the maximum number of UDP transmissions (first transmission plus retransmissions) of SIP messages. The range is 1 to 30. The default value is 7.
Web: Number of RTX Before Hot-Swap EMS: Proxy Hot Swap Rtx CLI: nb-of-rtx-b4-hot-swap **[HotSwapRtx]**	Defines the number of retransmitted INVITE/REGISTER messages before the call is routed (hot swap) to another Proxy/Registrar. The valid range is 1 to 30. The default value is 3.

Profile Parameters

The profile parameters are described in the table below.

Table A-28: Profile Parameters

Parameter	Description
IP Profile Table	
Web: IP Profile Settings EMS: Protocol Definition > IP Profile CLI: config voip > coders-and-profiles ip-profile **[IPProfile]**	This *para*meter table configures the IP Profile table. Each IP Profile ID includes a set of parameters (which are typically configured separately using their individual "global" parameters). The format of this parameter is as follows: [IPProfile] FORMAT IpProfile_Index = IpProfile_ProfileName, IpProfile_IpPreference, IpProfile_CodersGroupID, IpProfile_IsFaxUsed, IpProfile_JitterBufMinDelay, IpProfile_JitterBufOptFactor, IpProfile_IPDiffServ, IpProfile_SigIPDiffServ, IpProfile_SCE, IpProfile_RTPRedundancyDepth, IpProfile_RemoteBaseUDPPort, IpProfile_CNGmode, IpProfile_VxxTransportType, IpProfile_NSEMode, IpProfile_IsDTMFUsed, IpProfile_PlayRBTone2IP, IpProfile_EnableEarlyMedia, IpProfile_ProgressIndicator2IP, IpProfile_EnableEchoCanceller, IpProfile_CopyDest2RedirectNumber, IpProfile_MediaSecurityBehaviour, IpProfile_CallLimit, IpProfile_DisconnectOnBrokenConnection, IpProfile_FirstTxDtmfOption, IpProfile_SecondTxDtmfOption, IpProfile_RxDTMFOption, IpProfile_EnableHold, IpProfile_InputGain, IpProfile_VoiceVolume, IpProfile_AddIEInSetup, IpProfile_SBCExtensionCodersGroupID, IpProfile_MediaIPVersionPreference, IpProfile_TranscodingMode, IpProfile_SBCAllowedCodersGroupID, IpProfile_SBCAllowedCodersMode, IpProfile_SBCMediaSecurityBehaviour, IpProfile_SBCRFC2833Behavior, IpProfile_SBCAlternativeDTMFMethod, IpProfile_SBCAssertIdentity, IpProfile_AMDSensitivityParameterSuit, IpProfile_AMDSensitivityLevel, IpProfile_AMDMaxGreetingTime, IpProfile_AMDMaxPostSilenceGreetingTime, IpProfile_SBCDiversionMode, IpProfile_SBCHistoryInfoMode, IpProfile_EnableQSIGTunneling, IpProfile_SBCFaxCodersGroupID, IpProfile_SBCFaxBehavior, IpProfile_SBCFaxOfferMode, IpProfile_SBCFaxAnswerMode; [\IPProfile] For example: IPProfile 1 = ITSP, 1, 0, 0, 10, 10, 46, 40, 0, 0, 0, 0, 2, 0, 0, 0, 0, -1, 1, 0, 0, -1, 1, 4, -1, 1, 1, 0, 0, , -1, 0, 0, -1, 0, 0, 0, 0, -1, 0, 8, 300, 400, -1, -1; **Notes:** You can configure up to nine IP Profiles (i.e., indices 1 through 9). To use the settings of the corresponding "global" parameter, enter the value -1 (or in the Web interface, the option 'Not Configured'). For a detailed description of each parameter, see its corresponding global parameter: The parameter IpPreference determines the priority of the IP Profile (1 to 20, where 20 is the highest preference). If both IP and Tel Profiles apply to the same call, the coders and common parameters (i.e., parameters configurable in both IP and Tel Profiles) of the preferred profile are applied to that call. If the Tel and IP Profiles are identical, the Tel Profile parameters take precedence. The parameter CallLimit defines the maximum number of concurrent calls allowed for that Profile. If the Profile is set to some limit, the device maintains the number of concurrent calls (incoming and outgoing) pertaining to the specific Profile. A limit value of **[-1**] indicates that there is no limitation on calls (default). A limit value of **[0]** indicates that all calls are rejected. When the number of concurrent calls is equal to the limit, the device rejects any new incoming and outgoing calls

Parameter	Description
	pertaining to that profile. IP Profiles can also be used when operating with a Proxy server (set the parameter AlwaysUseRouteTable to 1). For a description of using *ini* file table parameters, see 'Configuring ini File Table Parameters' on page 54.

Channel Parameters

This subsection describes the device's channel parameters.

RTP, RTCP and T.38 Parameters

The RTP, RTCP and T.38 parameters are described in the table below.

Table A-29: RTP/RTCP and T.38 Parameters

Parameter	Description
Web: RTP Base UDP Port EMS: Base UDP Port **[BaseUDPport]**	Defines the lower boundary of the UDP port used for RTP, RTCP (RTP port + 1) and T.38 (RTP port + 2). For example, if the Base UDP Port is set to 6000, then one channel may use the ports RTP 6000, RTCP 6001, and T.38 6002, while another channel may use RTP 6010, RTCP 6011, and T.38 6012, and so on. The range of possible UDP ports is 6,000 to 64,000. The default base UDP port is 6000. Once this parameter is configured, the UDP port range (lower to upper boundary) is calculated as follows: BaseUDPport to (BaseUDPport + 329*10) **Notes:** For this parameter to take effect, a device reset is required. Once this parameter is configured, the UDP port range (lower to upper boundary) is calculated as follows: BaseUDPport to BaseUDPport + 4000*10 The UDP ports are allocated randomly to channels. You can define a UDP port range per Media Realm (see Configuring Media Realms on page 105). If RTP Base UDP Port is not a factor of 10, the following message is generated: 'invalid local RTP port'. For more information on the default RTP/RTCP/T.38 port allocation, refer to the *Product Reference Manual*.
EMS: No Op Enable CLI: no-operation-enable **[NoOpEnable]**	Enables the transmission of RTP or T.38 No-Op packets. **[0]** = Disable (default) **[1]** = Enable This mechanism ensures that the NAT binding remains open during RTP or T.38 silence periods.
EMS: No Op Interval **[NoOpInterval]**	Defines the time interval in which RTP or T.38 No-Op packets are sent in the case of silence (no RTP/T.38 traffic) when No-Op packet transmission is enabled. The valid range is 20 to 65,000 msec. The default is 10,000. **Note:** To enable No-Op packet transmission, use the NoOpEnable parameter.
EMS: No Op Payload Type CLI: no-operation-interval **[RTPNoOpPayloadType]**	Defines the payload type of No-Op packets. The valid range is 96 to 127 (for the range of Dynamic RTP Payload Type for all types of non hard-coded RTP Payload types, refer to RFC 3551). The default value is 120. **Note:** When defining this parameter, ensure that it doesn't cause collision with other payload types.

Parameter	Description
CLI: rtcp-act-mode **[RTCPActivationMode]**	Disables RTCP traffic when there is no RTP traffic. This feature is useful, for example, to stop RTCP traffic that is typically sent when calls are put on hold (by an INVITE with 'a=inactive' in the SDP). **[0]** Active Always = RTCP is active even during inactive RTP periods, i.e., when the media is in 'recvonly' or 'inactive' mode. (default) **[1]** Inactive Only If RTP Inactive = No RTCP is sent when RTP is inactive.

SBC Parameters

The SBC parameters are described in the table below.

Table A-30: SBC Parameters

Parameter	Description
Web: Enable SBC EMS: Enable SBC CLI: enable-sbc **[EnableSBCApplication]**	Enables the Session Border Control (SBC) application. **[0]** Disable (default) **[1]** Enable **Notes:** For this parameter to take effect, a device reset is required. In addition to enabling this parameter, the number of maximum SBC/IP-to-IP sessions must be defined in the Software Upgrade Key.
Web: Allow Unclassified Calls CLI: unclassified-calls **[AllowUnclassifiedCalls]**	Determines whether calls (incoming packets) that cannot be classified (i.e. classification process fails) into a Source IP Group (in the Classification table) are either rejected or processed. **[0]** Reject = the call is rejected if classification fails. **[1]** Allow = if classification fails, the incoming packet is assigned to the default IP Group of the default SRD (and the call is subsequently processed). (Default.)
Web: SBC No Answer Timeout CLI: sbc-no-arelt-timeout **[SBCAlertTimeout]**	Defines the timeout (in seconds) for SBC outgoing (outbound IP routing) SIP INVITE messages. If the called IP party does not answer the call within this user-defined interval, the device disconnects the session. The device starts the timeout count upon receipt of a SIP 180 Ringing response from the called party. If no other SIP response (for example, 200 OK) is received thereafter within this timeout, the call is released. The valid range is 0 to 3600 seconds. the default is 600.
Web: SBC Max Forwards Limit **[SBCMaxForwardsLimit]**	Defines the Max-Forwards SIP header value. The Max-Forwards header is used to limit the number of servers (such as proxies) that can forward the SIP request. The Max-Forwards value indicates the remaining number of times this request message is allowed to be forwarded. This count is decremented by each server that forwards the request. This parameter affects the Max-Forwards header in the received message as follows: If the received header's original value is 0, the message is not passed on and is rejected. If the received header's original value is less than this parameter's value, the header's value is decremented before being sent on. If the received header's original value is greater than the parameter's value, the header's value is replaced by the user-defined parameter's value. The valid value range is 1-70. The default is 10.
Web: Minimum Session-Expires CLI: min-session-expires **[SBCMinSE]**	Defines the minimum amount of time (in seconds) between session refresh requests in a dialog before the session is considered timed out. This value is conveyed in the SIP Min-SE header. The valid range is 0 (default) to 1,000,000 (where 0 means that the device does not limit Session-Expires).
Web/EMS: Handle P-Asserted-	Determines the device's privacy handling of the P-Asserted-Identity

Parameter	Description
Identity CLI: p-assert-id **[SBCAssertIdentity]**	header. This indicates how the outgoing SIP message asserts identity. **[0]** Don't Care (default) = P-Asserted Identity header is not affected. **[1]** Add P-Asserted-Identity Header = Adds a P-Asserted-Identity header. The header's values are taken from the source URL. **[2]** Remove P-Asserted-Identity Header = Removes the P-Asserted-Identity header. **Notes:** This parameter affects only the initial INVITE request. The configuration of privacy handling in the IP Group table takes precedence over the settings of this global parameter. If in the IP Group this parameter is set to 'Don't care', then the settings of this global parameter is used. If this global parameter and the IP Group are set to 'Don't care', the device uses the same P-Asserted-Identity header (if present) in the incoming message for the outgoing message. This parameter can also be configured per IP Profile, using the IPProfile parameter (see 'Configuring IP Profiles' on page 137).
Web: Keep original user in Register **[SBCKeepContactUserinRegister]**	Determines whether the device replaces the Contact user with a unique Contact user in the outgoing message in response to a REGISTER request. **[0]** Disable = (default) The device replaces the original Contact user with a unique Contact user, for example: Received Contact: <sip:123@domain.com> Outgoing (unique) Contact: <sip:FEU1_7_1@SBC> **[1]** Enable = The original Contact user is retained and used in the outgoing REGISTER request. **Note:** This parameter is applicable only to REGISTER messages received from USER IP Groups and that are sent to SERVER IP Groups.
CLI: sbc-refer-bhvr **[SBCReferBehavior]**	Determines the device's handling of REFER requests. **[0]** = Refer-To header is unchanged (default). **[1]** = Uses the database for Refer-To as described below. When enabled, the device handles REFERs as follows: 1 Before passing on the REFER request, the device changes the host part to the device's IP address and adds a special prefix ("T~&R_") to the Contact user part. The incoming INVITE is identified as a REFER-resultant INVITE according to the special prefix. The device replaces the host part (in the Request-URI) with the host from the REFER contact. The prefix ("T~&R_") remains in the user part for regular classification, manipulation, and routing. The special prefix can be used for specific routing rules for REFER-resultant INVITES. The prefix is removed before the resultant INVITE is sent to the destination.
CLI: sbc-xfer-prefix **[SBCXferPrefix]**	When the SBCReferBehavior is set to 1, the device, while interworking the SIP REFER message, adds the prefix "T~&R-" to the user part of the URI in the Refer-To header. After this, the device can receive an INVITE with such a prefix (the INVITE is sent by the UA that receives the REFER message or 302 response). If the device receives an INVITE with such a prefix, it replaces the prefix with the value defined for the SBCXferPrefix parameter.

Parameter	Description
	The default value is empty (""). **Note:** This feature is also applicable to 3xx redirect responses. The device adds the prefix "T~&R-" to the URI user part in the Contact header if the SBC3xxBehavior parameter is set to 1.
CLI: sbc-3xx-bhvt **[SBC3xxBehavior]**	Determines the device's handling of SIP 3xx responses. When enabled, the device handles SIP redirections between different subnets. This is required where the new address provided by the redirector (Redirect sever) may not be reachable by the far-end user (FEU) located in another subnet. For example, a far-end user (FEU) in the WAN sends a SIP request via the device to a Redirect server in the LAN, and the Redirect server replies with a SIP 3xx response to a PBX in the LAN in the Contact header. If the device sends this response as is (i.e., with the original Contact header), the FEU is unable to reach the new destination. **[0]** (default) = The device sends the received SIP 3xx response without changing the Contact header (transparent handling). **[1]** = The device changes the URI in the Contact header of the received SIP 3xx response to its own URI and adds a special user prefix ("T~&R_"), which is then sent to the FEU. The FEU then sends a new INVITE to the device, which the device then sends to the correct destination. **Notes:** When this parameter is changed from 1 to 0, new 3xx Contact headers remain unchanged. However, requests with the special prefix continue using the device's database to locate the new destination. Only one database entry is supported for the same host, port, and transport combination. For example, the following URLs cannot be distinguished by the device: sip:10.10.10.10:5060;transport=tcp;param=a sip:10.10.10.10:5060;transport=tcp;param=b The database entry expires two hours after the last use. The maximum number of destinations (i.e., database entries) is 50. For more information on SIP 3xx Redirect response handling, see 'Handling SIP 3xx Redirect Responses' on page 162.
Web: Registration Mode **[IPGroup_RegistrationMode]**	Defines the registration mode for an IP Group: **[0]** User initiates registrations (default) **[1]** SBC initiate registrations (works only with User Info file) = Used when the device serves as a client (e.g., with an IP PBX) **[2]** No registrations needed = The device adds users to its database in active state. **Note:** This is an IP Group table parameter.
Web: Authentication Mode **[PGroup_AuthenticationMode]**	Defines the authentication mode: **[0]** User Authenticates (default) = The device does not handle the authentication, but simply passes the authentication messages between the SIP user agents. **[1]** SBC Authenticates (as client) = The device authenticates as a client. It receives the 401/407 response from the proxy requesting for authentication. The device sends the proxy the authorization credentials (i.e., user name and password) according to one of the following: 1) account defined in the Account table (only if authenticating SERVER-type IP Group), 2) global username and password parameters (only if authenticating SERVER-type IP Group), 3) User Information file, or 4) sends request to users

Parameter	Description
	requesting credentials (only if authenticating USER-type IP Group). **[2]** SBC Authenticates (as server) = The device authenticates as a server (using the User Information file). **Note:** This is an IP Group table parameter.
Web: Authentication Method List **[IPGroup_MethodList]**	Defines the SIP methods that the device must challenge. Multiple entries are separated by the forward slash "/". If none are defined (default), no methods are challenged. **Note:** This is an IP Group table parameter.
Web: Lifetime of the nonce in seconds CLI: lifetime-of-nonce **[AuthNonceDuration]**	Defines the lifetime (in seconds) that the current nonce is valid for server-based authentication. The device challenges a message that attempts to use a server nonce beyond this period. This parameter is used to provide replay protection (i.e., ensures that old communication streams are not used in replay attacks). The valid value range is 30 to 600. The default value is 300.
Web: Authentication Challenge Method CLI: auth-chlng-mthd **[AuthChallengeMethod]**	Defines the type of server-based authentication challenge. **[0]** 0 = Send SIP 401 "Unauthorized" with a WWW-Authenticate header as the authentication challenge response. (default) **[1]** 1 = Send SIP 407 "Proxy Authentication Required" with a Proxy-Authenticate header as the authentication challenge response.
Web: Authentication Quality of Protection CLI: auth-qop **[AuthQOP]**	Defines the authentication and integrity level of quality of protection (QOP) for digest authentication offered to the client. When the device challenges a SIP request (e.g., INVITE), it sends a SIP 401 response with the Authorization header containing the 'qop' parameter indicating the QoP level of the message to be authenticated. In response, the SBC client needs to send the device another INVITE with the MD5 hash of the INVITE message and indicates its auth or auth-int support. **[0]** Auth = The device sends 'qop=auth' in the SIP response, requesting authentication (i.e., validates user by checking user name and password). This option does not authenticate the message body (i.e., SDP). **[1]** auth-int = = The device sends 'qop=auth-int' in the SIP response, indicating required authentication and authentication with integrity (e.g., checksum). This option restricts the client to authenticating the entire SIP message, including the body, if present. **[2]** Auth-Int and Auth = The device sends 'qop=auth, auth-int' in the SIP response, indicating either authentication or integrity (default). This enables the client to choose auth or auth-int. If the client chooses auth-int, the body is included in the authentication. If the client chooses auth, then the body is not authenticated. (default)
Web: SBC User Registration Time CLI: sbc-usr-reg-time **[SBCUserRegistrationTime]**	Defines the duration (in seconds) of the periodic registrations between the user and the device (the device responds with this value to the user). When set to 0, the device does not change the Expires header's value received in the user's REGISTER request. If no Expires header is received in the REGISTER message and the SBCUserRegistrationTime parameter is set to 0, then by default, the Expires header's value is set to 180 seconds. The valid range is 0 to 2,000,000 seconds. The default is 0. **Note:** For this parameter to take effect, a device reset is required.
Web: SBC Proxy Registration Time CLI: sbc-prxy-reg-time **[SBCProxyRegistrationTime]**	Defines the duration (in seconds) for which the user is registered in the proxy database (after the device forwards the REGISTER message). When set to 0, the device sends the Expires header's value as received from the user to the proxy.

Parameter	Description
	The valid range is 0 to 2,000,000 seconds. The default is 0.
Web: SBC Survivability Registration Time CLI: sbc-surv-reg-time **[SBCSurvivabilityRegistrationTime]**	Defines the duration of the periodic registrations between the user and the device, when the device is in survivability state (i.e., when REGISTER requests cannot be forwarded to the proxy and are terminated by the device). When set to 0, the device uses the value set by the SBCUserRegistrationTime parameter for the device's response. The valid range is 0 to 2,000,000 seconds. The default is 0.

[SBCEnableAASTRASurvivabilityNotice]	Enables the device to notify Aastra IP phones that it is currently operating in Survivability mode. When this occurs, the Aastra IP phones display the message, "Stand Alone Mode" on their LCD screens. Survivability mode occurs when connectivity with the WAN fails and as a result, the device enables communication between IP phone users within the LAN enterprise.
	[0] = Disable
	[1] = Enable
	When this feature is enabled and the SBC device is in Survivability mode, it responds to SIP REGISTER messages from the IP phones with a SIP 200 OK containing the following XML body:
	`Content-Type: application/xml`
	`<?xml version="1.0" encoding="utf-8"?>`
	`<LMIDocument version="1.0">`
	`<LocalModeStatus>`
	`<LocalModeActive>true</LocalModeActive>`
	`<LocalModeDisplay>StandAlone Mode</LocalModeDisplay>`
	`</LocalModeStatus>`
	`</LMIDocument>`

Web: SBC GRUU Mode CLI: sbc-gruu-mode **[SBCGruuMode]**	Determines the Globally Routable User Agent (UA) URI (GRUU) support, according to RFC 5627. **[0]** None = No GRUU is supplied to users. **[1]** As Proxy = The device provides same GRUU types as the proxy provided the device's GRUU clients. (default) **[2]** Temporary only = Supply only temporary GRUU to users. (Currently not supported.) **[3]** Public only = The device provides only public GRUU to users. **[4]** Both = The device provides temporary and public GRUU to users. (Currently not supported.) This parameter allows the device to act as a GRUU server for its SIP UA clients, providing them with public GRUU's, according to RFC 5627. The public GRUU provided to the client is denoted in the SIP Contact header parameters, "pub-gruu". Public GRUU remains the same over registration expirations. On the other SBC leg communicating with the Proxy/Registrar, the device acts as a GRUU client. The device creates a GRUU value for each of its registered clients, which is mapped to the GRUU value received from the Proxy server. In other words, the created GRUU value is only used between the device and its clients (endpoints). `Public-GRUU: sip:userA@domain.com;gr=unique-id`

Parameter	Description
Enable Bye Authentication **[SBCEnableByeAuthentication]**	Enables authenticating a SIP BYE request before disconnecting the call. This feature prevents, for example, a scenario in which the SBC SIP client receives a BYE request from a third-party imposer assuming the identity of a participant in the call and as a consequence, the call between the first and second parties is inappropriately disconnected. **[0]** Disable (default) **[1]** Enable = The device forwards the SIP authentication response (for the BYE request) to the request sender and waits for the user to authenticate it. The call is disconnected only if the authenticating server responds with a 200 OK.
[SBCExtensionsProvisioningMode]	Enables SBC user registration for interoperability with BroadSoft's BroadWorks server, to provide call survivability in case of connectivity failure with the BroadWorks server. **[0]** = Normal processing of REGISTER messages (default). **[1]** = Registration method for BroadWorks server. In a failure scenario with BroadWorks, the device acts as a backup SIP proxy server, maintaining call continuity between the enterprise LAN users (subscribers). **Note:** For a detailed description of this feature, see 'Auto-Provisioning of Subscriber-Specific Information for BroadWorks Server' on page 165.
Web: SBC Direct Media **[SBCDirectMedia]**	Enables the No Media Anchoring feature (i.e., direct media) for all SBC calls. No Media Anchoring uses SIP signaling capabilities without handling the RTP/SRTP (media) flow between remote SIP user agents (UA). The RTP packets do not traverse the device, instead, the two SIP UAs establish a direct RTP/SRTP flow between one another. Signaling continues to traverse the device with minimal intermediation and involvement to enable certain SBC abilities such as routing **[0]** Disable = All SRD calls via SBC are not direct media - internal SRD calls are according to SRD configuration (default). **[1]** Enable = All SBC calls use the No Media Anchoring feature (i.e., direct media). **Notes:** For more information on No Media Anchoring, see 'No Media Anchoring' on page 156. When No Media Anchoring is enabled: Manipulation is not done on SDP data (offer/answer transaction) such as ports and IP addresses. Opening voice channels and allocation of IP media ports are not required. The Coder Restriction feature (Allowed Coders List) operates simultaneously with No Media Anchoring calls. Restricted coders are removed from the SDP offer message. No Media Anchoring is typically implemented in the following scenarios: SBC device is located in the LAN. Calls between two SIP UAs in the same LAN and signals are sent to a SIP proxy server that is located in the WAN. SBC device does not do NAT traversal (for media) and all the users are in the same domain. The benefits of implementing the No Media Anchoring feature includes the following: saves network bandwidth, reduces CPU usage (no RTP/SRTP handling), and avoids interference in SDP negotiation

Parameter	Description
	and header manipulation on RTP/SRTP. The process for handling the No Media Anchoring feature is as follows: Identifying a No Media Anchoring call according to configuration and the call's properties (such as source, destination, IP Group, and SRD). Handing the identified No Media Anchoring call. You can enable No Media Anchoring per SRD (using the IntraSRDMediaAnchoring parameter), whereby calls between two UAs that pertain to the same SRD (source and destination) are handled as a No Media Anchoring (direct media) call. Chosen configuration can't handle call from any UA to a foreign UA (vice versa) but both UAs belong to the same SRD and the parameter IntraSRDMediaAnchoring for that specific SRD is > 0. When this parameter is disabled, No Media Anchoring calls between two UAs that belong to separate SRDs cannot be configured. No Media Anchoring calls between two UAs that belong to the same SRD is configurable only (in this case).
CLI: media-sec-bhvior **[IpProfile_SBCMediaSecurityBehaviour]**	Determines the transcoding method between SRTP and RTP. The device can also enforce SBC legs to use SRTP\RTP, using the IP Profile parameter SBCMediaSecurityBehaviour: **[0]** As is (default): no special handling for RTP\SRTP is done. **[1]** SRTP: SBC legs negotiate only SRTP media lines, and RTP media lines are removed from the incoming SDP offer\answer. **[2]** RTP: SBC legs negotiate only RTP media lines, and SRTP media lines are removed from the incoming offer\answer. **[3]** Both: each offer\answer is extended (if not already) to two media lines - one RTP and the other SRTP. If two SBC legs (after offer\answer negotiation) use different security types (i.e., one RTP and the other SRTP), then the device performs RTP-SRTP transcoding. To transcode between RTP and SRTP, the following prerequisites must be met: At least one supported SDP "crypto" attribute and parameters EnableMediaSecurity must be set to 1 If one of the above transcoding prerequisites is not met, then: any value other than "As is" is discarded. if the incoming offer is SRTP, force transcoding, coder transcoding, and DTMF extensions are not applied. Transcoding between RTP and SRTP requires alloctaion of one DSP. SRTP-to-SRTP transcoding does not require any DSP allocation. **Note:** This parameter can only be configured as an IP Profile, using the IPProfile parameter (see 'Configuring IP Profiles' on page 137).
[IpProfile_SBCRFC2833Behavior]	Determines the RFC 2833 SDP offer\answer negotiation. **[0]** As is = The device does not intervene in the RFC 2833 negotiation. (default) **[1]** Extend = Each outgoing offer\answer includes RFC 2833 in the offered SDP (the device adds RFC 2833 only if the incoming offer does not include RFC 2833). **[2]** Disallow = The device removes RFC 2833 from the incoming offer. **Note:** This parameter can only be configured as an IP Profile, using the IPProfile parameter (see Configuring IP Profiles on page 137).
[IpProfile_SBCAlternativeDTMF	The device's first priority for DTMF method at each leg is RFC 2833. Therefore, if a specific leg negotiates RFC 2833 successfully, then the

Parameter	Description
Method]	chosen DTMF method for this leg is RFC 2833. For legs where RFC 2833 is not negotiated successfully, the device uses this parameter to determine the chosen DTMF method for the leg. **[0]** = Don't care - the device does not attempt to interwork any special DTMF method. (default) **[1]** = In Band **[2]** = INFO, Cisco **[3]** = INFO, Nortel **[4]** = INFO, Korea **Note:** This parameter can only be configured as an IP Profile, using the IPProfile parameter (see Configuring IP Profiles on page 137).
Web: Diversion Mode **[IpProfile_SBCDiversionMode]**	Determines the device's handling of the SIP Diversion header. For more information on interworking of the History-Info and Diversion headers, see 'Interworking SIP Diversion and History-Info Headers' on page 164. **[0]** Don't Care = Diversion header is not handled. (default) **[1]** Add = History-Info header converted to a Diversion header. **[2]** Remove = Removes the Diversion header and the conversion to the History-Info header depends on the settings of the SBCHistoryInfoMode parameter. **Note:** This parameter can only be configured as an IP Profile, using the IPProfile parameter (see 'Configuring IP Profiles' on page 137).
Web: History Info Mode **[IpProfile_SBCHistoryInfoMode]**	Determines the device's handling of the History-Info header. For more information on interworking of the History-Info and Diversion headers, see 'Interworking SIP Diversion and History-Info Headers' on page 164. **[0]** Don't Care = History-Info header is not handled. (default) **[1]** Add = Diversion header converted to a History-Info header. **[2]** Remove = History-Info header removed from the SIP dialog and the conversion to the Diversion header depends on the settings of the SBCDiversionMode parameter. **Note:** This parameter can only be configured as an IP Profile, using the IPProfile parameter (see 'Configuring IP Profiles' on page 137).

Parameter	Description
Web: Allowed Coders Mode **[IpProfile_SBCAllowedCodersMode]**	Determines the mode of the Allowed Coders feature for both SBC legs. **[0]** Restriction = In the incoming SDP offer, the device uses only coders that are also listed in the Allowed Coders Group; the rest are removed from the SDP offer (i.e., only coders common between SDP offered coders and Allowed Coders Group are used). **[1]** Preference = The device re-arranges the priority (order) of the coders in the incoming SDP offer according to their order of appearance in the Allowed Coders Group list. (This option also retains all the coders received in the SDP offer.) **[2]** Restriction and Preference = Performs both Restriction and Preference. **Notes:** If the AllowedCodersGroup parameter is set to None, then this parameter is not applicable. This parameter can only be configured as an IP Profile, using the IPProfile parameter (see 'Configuring IP Profiles' on page 137). To select the Allowed Coders Group ID, use the AllowedCodersGroup parameter. For more information on the Allowed Coders feature, see 'Coder Restrictions Control' on page 157.
SBC Fax Coders Group ID **[IpProfile_SBCFaxCodersGroupID]**	Selects the supported fax coders (Coders Group ID) for fax negotiation. Coders Groups are configured in the Coders Group Settings table. **Note:** This parameter can only be configured as an IP Profile, using the IPProfile parameter (see 'Configuring IP Profiles' on page 137).
SBC Fax Behavior **[IpProfile_SBCFaxBehavior]**	Defines the negotiation method for fax offer: **[0]** = Pass fax transparently, without interference (default). **[1]** = Handle fax according to fax settings in the IP Profile for all offer-answer transactions (including the initial INVITE). **[2]** = Handle fax according to fax settings in the IP Profile for all re-INVITE offer-answer transactions (except for initial INVITE). **Note:** This parameter can only be configured as an IP Profile, using the IPProfile parameter (see 'Configuring IP Profiles' on page 137).
SBC Fax Offer Mode **[IpProfile_SBCFaxOfferMode]**	Defines the coders included in the outgoing SDP offer (sent to the called "fax"). **[0]** All = Use only (and all) the coders of the selected Coders Group ID configured using the SBCFaxCodersGroupID parameter. (Default) **[1]** Single = Use only one coder. If a coder in the incoming offer (from the calling "fax") matches a coder in the SBCFaxCodersGroupID, then the device uses this coder. If no match exists, then the device uses the first coder listed in the Coders Group ID (SBCFaxCodersGroupID). **Note:** This parameter can only be configured as an IP Profile, using the IPProfile parameter (see 'Configuring IP Profiles' on page 137).

Parameter	Description
SBC Fax Answer Mode **[IpProfile_SBCFaxAnswerMode]**	Defines the coders included in the outgoing SDP answer (sent to the calling "fax"). **[0]** All = Use matched coders between the incoming offer coders (from the calling "fax") and the coders of the selected Coders Group ID (configured using the SBCFaxCodersGroupID parameter). **[1]** Single = Use only one coder. If the incoming answer (from the called "fax") includes a coder that matches a coder match between the incoming offer coders (from the calling "fax") and the coders of the selected Coders Group ID (SBCFaxCodersGroupID, then the device uses this coder. If no match exists, the device uses the first listed coder of the matched coders between the incoming offer coders (from the calling "fax") and the coders of the selected Coders Group ID. (Default) **Note:** This parameter can only be configured as an IP Profile, using the IPProfile parameter (see 'Configuring IP Profiles' on page 137).
Web: Enable SBC Client Forking **[IPGroup_EnableSBCClientForking]**	Enables call forking for USER-type IP Groups. **[0]** No (default) **[1]** Yes = The device forks INVITE messages (to up to five separate SIP outgoing legs). **Note:** This parameter can only be configured for an IP Group, using the IPGroup parameter.
[EnableSBCMediaSync]	Enables SBC media synchronization process for calls established from SIP forking that is initiated by external proxy servers. It is possible that a call is established with the media not synchronized between the SBC legs. Media synchronization resolves this issue. **[0]** Disable **[1]** Enable (default)
Admission Control Table	
Web: Admission Control EMS: Call Admission Control CLI: configure voip > sbc sbc-admission-control **[SBCAdmissionControl]**	This *para*meter table defines limitations on the number of allowed concurrent calls (SIP dialogs). This is useful for controlling bandwidth utilization between Voice and Data traffic. The format of this parameter is as follows: [SBCAdmissionControl] FORMAT SBCAdmissionControl_Index = SBCAdmissionControl_LimitType, SBCAdmissionControl_IPGroupID, SBCAdmissionControl_SRDID, SBCAdmissionControl_RequestType, SBCAdmissionControl_RequestDirection, SBCAdmissionControl_Limit, SBCAdmissionControl_LimitPerUser, SBCAdmissionControl_Rate, SBCAdmissionControl_MaxBurst; [\SBCAdmissionControl] For example, the below configuration allows a maximum of 10 concurrent SIP INVITEs for IP Group 1: SBCAdmissionControl 1 = 0, 1, -1, 1, 0, 10, -1, 0, 0; **Notes:** For a detailed description of the table's individual parameters and for configuring the table using the Web interface, see 'Configuring Admission Control' on page 173. For a description on configuring ini file table parameters, see 'Configuring ini File Table Parameters' on page 54.
Allowed Audio Coders Table	
Web: Allowed Audio Coders	This *para*meter table allows you to define up to 5 Allowed Coders

Parameter	Description
CLI: configure voip > sbc allowed-coders-group AllowedCodersGroup0 **[AllowedCodersGroup0]** **[AllowedCodersGroup1]** **[AllowedCodersGroup2]** **[AllowedCodersGroup3]** **[AllowedCodersGroup4]**	Groups, each with up to 10 coders. The Allowed Coders Group determines the coders that can be used for a specific SBC leg. Coders excluded from the Allowed Coders Group are removed from the SDP offer (only coders common between SDP offered coders and Allowed Coders are used). In addition, coders defined in top entries in the Allowed Coders Group are assigned higher priority than those entered in lower entries. [AllowedCodersGroupx] FORMAT AllowedCodersGroup_Index = AllowedCodersGroup_Name; [\AllowedCodersGroup] Where, AllowedCodersGroupx = Allowed Coders Group index (0-4). Index = Coder index number per group (0-9). Name = Coder name. For supported coders, see the CodersGroup parameter. For example, below represents two configured Allowed Coders Groups, 0 and 1. Group 0 has two coders; Group 1 has one coder. The highest priority coder is G.723.1. [AllowedCodersGroup0] FORMAT AllowedCodersGroup0_Index = AllowedCodersGroup0_Name; AllowedCodersGroup0 0 = g7231; AllowedCodersGroup0 1 = g711Alaw64k; [\AllowedCodersGroup0] [AllowedCodersGroup1] FORMAT AllowedCodersGroup1_Index = AllowedCodersGroup0_Name; AllowedCodersGroup1 0 = g711Ulaw64k; [\AllowedCodersGroup1] **Notes:** The Allowed Coders table is performed on audio media only. Allowed Coder Groups can be assigned to IP Profiles (see 'Configuring IP Profiles' on page 137). For configuring the table using the Web interface, see 'Configuring Allowed Coder Groups' on page 175. For a description on configuring *ini* file table parameters, see 'Configuring ini File Table Parameters' on page 54.
Message Policy Table	
Web: Message Policy Table CLI: configure voip > sbc message-policy **[MessagePolicy]**	This *para*meter table configures SIP message policy rules. The format of this parameter is as follows: [MessagePolicy] FORMAT MessagePolicy_Index = MessagePolicy_Policy, MessagePolicy_MaxMessageLength, MessagePolicy_MaxHeaderLength, MessagePolicy_MaxBodyLength, MessagePolicy_MaxNumHeaders, MessagePolicy_MaxNumBodies, MessagePolicy_SendRejection, MessagePolicy_MethodListType, MessagePolicy_MethodList, MessagePolicy_BodyListType, MessagePolicy_BodyList; [/MessagePolicy]
Classification Table	
Web: Classification Table EMS: SBC Classification	This *para*meter table configures the Classification table. This table classifies the incoming SIP INVITE to a Source IP Group. The format

Parameter	Description
CLI: configure voip > sbc routing classification **[Classification]**	of this parameter is as follows: [Classification] FORMAT Classification_Index = Classification_MessageCondition, Classification_SrcSRDID, Classification_SrcAddress, Classification_SrcPort, Classification_SrcTransportType, Classification_SrcUsernamePrefix, Classification_SrcHost, Classification_DestUsernamePrefix, Classification_DestHost, Classification_ActionType, Classification_SrcIPGroupID; [\Classification] For example: Classification 1 = 1, , 10.8.6.15, 5060, 2, *, *, *, *, 1, 4; **Notes:** This table can include up to 20 indices (where 0 is the first index). For a detailed description of the table's individual parameters and for configuring the table using the Web interface, see 'Configuring Classification Table' on page 178. For a description on configuring *ini* file table parameters, see 'Configuring ini File Table Parameters' on page 54.
Condition Table	
Web: Condition Table CLI: configure voip > sbc routing condition-table **[ConditionTable]**	Configures Conditions for SIP messages and supports the same syntax used in the SIP Message Manipulation table. These Condition rules are later assigned to Classification rules in the Classification table for enhancing the process for classifying an incoming SIP dialog to an IP Group. [ConditionTable] FORMAT ConditionTable_Index = ConditionTable_Condition, ConditionTable_Description; [\ConditionTable]
SBC IP-to-IP Routing Table	
Web: IP2IP Routing Table EMS: IP to IP Routing CLI: configure voip > sbc routing ip2ip-routing **[IP2IPRouting]**	This *para*meter table configures the SBC IP-to-IP Routing table for routing received SIP messages such as INVITE messages to an IP destination. The format of this parameter is as follows: [IP2IPRouting] FORMAT IP2IPRouting_Index = IP2IPRouting_SrcIPGroupID, IP2IPRouting_SrcUsernamePrefix, IP2IPRouting_SrcHost, IP2IPRouting_DestUsernamePrefix, IP2IPRouting_DestHost, IP2IPRouting_RequestType, IP2IPRouting_MessageCondition, IP2IPRouting_DestType, IP2IPRouting_DestIPGroupID, IP2IPRouting_DestSRDID, IP2IPRouting_DestAddress, IP2IPRouting_DestPort, IP2IPRouting_DestTransportType, IP2IPRouting_AltRouteOptions, IP2IPRouting_CostGroup; [\IP2IPRouting] For example: IP2IPRouting 1 = 1, *, *, *, *, 3, 0, -1, -1, , 0, -1, 0,,; **Notes:** This table can include up to 120 indices (where 0 is the first index). For a specific routing rule to be effective, the matching characteristics must match. If no matching rule is located, the call is rejected. For a detailed description of the table's individual parameters and for configuring the table using the Web interface, see 'Configuring SBC IP2IP Routing' on page 183. For a description on configuring *ini* file table parameters, see

Parameter	Description
	'Configuring ini File Table Parameters' on page 54.
SBC Alternative Routing Reasons Table	
Web: SBC Alternative Routing Reasons EMS: Alternative Routing Reasons CLI: configure voip > sbc routing sbc-alternative-routing-reasons **[SBCAlternativeRoutingReasons]**	This *parameter* table configures the SBC Alternative Routing Reasons table. This table is used for alternative IP-to-IP routing (defined in the IP2IP Routing table). If 4xx, 5xx, or 6xx SIP responses are received as a result of outgoing SIP dialog-initiating methods (e.g., INVITE, OPTIONS, and SUBSCRIBE messages), the device re-sends the messages (to an alternative route) if the response is defined in this table and if there are alternative routes configured in the IP2IP Routing table. The format of this parameter is as follows: [SBCAlternativeRoutingReasons] FORMAT SBCAlternativeRoutingReasons_Index = SBCAlternativeRoutingReasons_ReleaseCause; [\SBCAlternativeRoutingReasons] For example: SBCAlternaiveRoutingReasons 0 = 403; SBCAlternativeRoutingReasons 1 = 404; **Notes:** This table can include up to five indices (where 0 is the first index). For a description on configuring *ini* file table parameters, see 'Configuring ini File Table Parameters' on page 54.
Message Manipulations Table	
Web: Message Manipulations EMS: Message Manipulations CLI: configure voip > sbc manipulations message-manipulations **[MessageManipulations]**	This parameter table defines manipulation rules for SIP header messages. The format of this parameter is as follows: [MessageManipulations] FORMAT MessageManipulations_Index = MessageManipulations_ManSetID, MessageManipulations_MessageType, MessageManipulations_Condition, MessageManipulations_ActionSubject, MessageManipulations_ActionType, MessageManipulations_ActionValue, MessageManipulations_RowRole; [\MessageManipulations] For example, the below configuration changes the user part of the SIP From header to 200: MessageManipulations 1 = 0, Invite.Request, , Header.From.Url.User, 2, 200, 0; **Notes:** This table can include up to 200 indices (where 1 is the first index). You must enclose a string in a single apostrophe. If you are using multiple strings, then the entire string must also be enclosed in double apostrophe, for example, "<sip:' + header.from.url.user + '@domain.com>'". For a detailed description of the table's individual parameters and for configuring the table using the Web interface, see 'Configuring Message Manipulations' on page 192. For a description on configuring ini file table parameters, see 'Configuring ini File Table Parameters' on page 54.

Parameter	Description
IP to IP Inbound Manipulation Table	
Web: IP to IP Inbound Manipulation EMS: IP to IP Inbound Manipulation CLI: configure voip > sbc manipulations ip-inbound-manipulation **[IPInboundManipulation]**	This *para*meter table configures the IP to IP Inbound Manipulation table. This table allows you to manipulate the SIP URI user part (source and/or destination) of the inbound SIP dialog message. The format of this parameter is as follows: **[IPInboundManipulation]** FORMAT IPInboundManipulation_Index = IPInboundManipulation_IsAdditionalManipulation, IPInboundManipulation_ManipulatedURI, IPInboundManipulation_ManipulationPurpose, IPInboundManipulation_SrcIPGroupID, IPInboundManipulation_SrcUsernamePrefix, IPInboundManipulation_SrcHost, IPInboundManipulation_DestUsernamePrefix, IPInboundManipulation_DestHost, IPInboundManipulation_RequestType, IPInboundManipulation_RemoveFromLeft, IPInboundManipulation_RemoveFromRight, IPInboundManipulation_LeaveFromRight, IPInboundManipulation_Prefix2Add, IPInboundManipulation_Suffix2Add; **[\IPInboundManipulation**] For example: IPInboundManipulation 1 = 0, 0, 0, -1, *, abc, *, *, 0, 0, 0, 255, , ; **Notes:** This table can include up to 100 indices. For SIP URI host name (source and destination) manipulations, you can also use the IP Group table. These host names are simply replaced with the names configured for the Source and Destination IP Groups respectively. For a detailed description of the table's individual parameters and for configuring the table using the Web interface, see 'Configuring IP-to-IP Inbound Manipulations' on page 195. For a description on configuring *ini* file table parameters, see 'Configuring ini File Table Parameters' on page 54.
IP to IP Outbound Manipulation Table	
Web: IP to IP Outbound Manipulation EMS: IP to IP Outbound Manipulation CLI: configure voip > sbc manipulations ip-outbound-manipulation **[IPOutboundManipulation]**	This *para*meter table configures the IP to IP Outbound Manipulation table. This table allows you to manipulate the SIP URI user part (source and/or destination) of the outbound SIP dialog message. The format of this parameter is as follows: [IPOutboundManipulation] FORMAT IPOutboundManipulation_Index = IPOutboundManipulation_IsAdditionalManipulation, IPOutboundManipulation_ManipulatedURI, IPOutboundManipulation_SrcIPGroupID, IPOutboundManipulation_DestIPGroupID, IPOutboundManipulation_SrcUsernamePrefix, IPOutboundManipulation_SrcHost, IPOutboundManipulation_DestUsernamePrefix, IPOutboundManipulation_DestHost, IPOutboundManipulation_RequestType, IPOutboundManipulation_RemoveFromLeft, IPOutboundManipulation_RemoveFromRight, IPOutboundManipulation_LeaveFromRight,

Parameter	Description
	IPOutboundManipulation_Prefix2Add, IPOutboundManipulation_Suffix2Add, IPOutboundManipulation_PrivacyRestrictionMode; [\IPOutboundManipulation] For example: IPOutboundManipulation 1 = 0, 0, 2, -1, *, *, *, *, 1, 3, 0, 255, , , 0; **Notes:** This table can include up to 100 indices (where 0 is the first index). Manipulated destination SIP URI user names are done on the following SIP headers: Request URI, To, and Remote-Party-ID (if exists). Manipulated source SIP URI user names are done on the following SIP headers: From, P-Asserted (if exists), P-Preferred (if exists), and Remote-Party-ID (if exists). For SIP URI host name (source and destination) manipulations, you can also use the IP Group'table. These host names are simply replaced with the names configured for the Source and Destination IP Groups respectively. For a detailed description of the table's individual parameters and for configuring the table using the Web interface, see 'Configuring IP-to-IP Outbound Manipulations' on page 198. For a description on configuring *ini* file table parameters, see 'Configuring ini File Table Parameters' on page 54.

Standalone Survivability Parameters

The Stand-alone Survivability (SAS) parameters are described in the table below.

Table A-31: SAS Parameters

Parameter	Description
Web: Enable SAS EMS: Enable CLI: enable-sas **[EnableSAS]**	Enables the Stand-Alone Survivability (SAS) feature. **[0]** Disable Disabled (default) **[1]** Enable = SAS is enabled When enabled, the device receives the registration requests from different SIP entities in the local network and then forwards them to the defined proxy. If the connection to the proxy fails ('Emergency Mode'), the device serves as a proxy by allowing calls internal to the local network. **Note:** For this parameter to take effect, a device reset is required.
Web: SAS Local SIP UDP Port EMS: Local SIP UDP CLI: sas-local-sip-udp-port **[SASLocalSIPUDPPort]**	Defines the local UDP port for sending and receiving SIP messages for SAS. The SIP entities in the local network need to send the registration requests to this port. When forwarding the requests to the proxy ('Normal Mode'), this port serves as the source port. The valid range is 1 to 65,534. The default value is 5080.
Web: SAS Default Gateway IP EMS: Default Gateway IP CLI: sas-default-gw-ip **[SASDefaultGatewayIP]**	Defines the Default Gateway used in SAS 'Emergency Mode'. When an incoming SIP INVITE is received and the destination Address-Of-Record is not included in the SAS database, the request is immediately sent to this default gateway. The address can be configured as an IP address (dotted-decimal notation) or as a domain name (up to 49 characters). You can also configure the IP address with a destination port, e.g., "10.1.2.3:5060". The default is a null string, i.e., the local IP address of the gateway.
Web: SAS Registration Time EMS: Registration Time CLI: sas-registration-time **[SASRegistrationTime]**	Defines the value of the SIP Expires header that is sent in a 200 OK response to an incoming REGISTER message when in SAS 'Emergency Mode'. The valid range is 0 (Analog) or 10 (Digital) to 2,000,000. The default value is 20.
Web: SAS Local SIP TCP Port EMS: Local SIP TCP Port CLI: sas-local-sip-tcp-port **[SASLocalSIPTCPPort]**	Defines the local TCP port used to send/receive SIP messages for the SAS application. The SIP entities in the local network need to send the registration requests to this port. When forwarding the requests to the proxy ('Normal Mode'), this port serves as the source port. The valid range is 1 to 65,534. The default value is 5080.
Web: SAS Local SIP TLS Port EMS: Local SIP TLS Port CLI: sas-local-sip-tls-port **[SASLocalSIPTLSPort]**	Defines the local TLS port used to send/receive SIP messages for the SAS application. The SIP entities in the local network need to send the registration requests to this port. When forwarding the requests to the proxy ('Normal Mode'), this port serves as the source port. The valid range is 1 to 65,534. The default value is 5081.
Web/EMS: Enable Record-Route CLI: record-route **[SASEnableRecordRoute]**	Determines whether the device's SAS application adds the SIP Record-Route header to SIP requests. This ensures that SIP messages traverse the device's SAS agent by including the SAS IP address in the Record-Route header. **[0]** Disable (default) **[1]** Enable The Record-Route header is inserted in a request by a SAS proxy to

Parameter	Description
	force future requests in the dialog session to be routed through the SAS agent. Each traversed proxy in the path can insert this header, causing all future dialogs in the session to pass through it as well. When this feature is enabled, the SIP Record-Route header includes the URI "lr" parameter, indicating loose routing, for example: `Record-Route: <sip:server10.biloxi.com;lr>`
Web: SAS Proxy Set EMS: Proxy Set CLI: sas-proxy-set **[SASProxySet]**	Defines the Proxy Set (index number) used in SAS Normal mode to forward REGISTER and INVITE requests from users that are served by the SAS application. The valid range is 0 to 5. The default value is 0 (i.e., default Proxy Set).
Web: Redundant SAS Proxy Set EMS: Redundant Proxy Set CLI: rdcy-sas-proxy-set **[RedundantSASProxySet]**	Defines the Proxy Set (index number) used in SAS Emergency mode for fallback when the user is not found in the Registered Users database. Each time a new SIP request arrives, the SAS application checks whether the user is listed in the registration database. If the user is located in the database, the request is sent to the user. If the user is not found, the request is forwarded to the next redundant SAS defined in the Redundant SAS Proxy Set. If that SAS Proxy IP appears in the Via header of the request, it is not forwarded (thereby, preventing loops in the request's course). If no such redundant SAS exists, the SAS sends the request to its default gateway (configured by the parameter SASDefaultGatewayIP). The valid range is -1 to 5. The default value is -1 (i.e., no redundant Proxy Set).
Web/EMS: SAS Block Unregistered Users CLI: sas-block-unreg-usrs **[SASBlockUnRegUsers]**	Determines whether the device rejects SIP INVITE requests received from unregistered SAS users. This applies to SAS Normal and Emergency modes. **[0]** Un-Block = Allow INVITE from unregistered SAS users (default). **[1]** Block = Reject dialog-establishment requests from un-registered SAS users.
CLI: sas-contact-replace **[SASEnableContactReplace]**	Enables the device to change the SIP Contact header so that it points to the SAS host and therefore, the top-most SIP Via header and the Contact header point to the same host. **[0]** (default) = Disable - when relaying requests, the SAS agent adds a new Via header (with the SAS IP address) as the top-most Via header and retains the original Contact header. Thus, the top-most Via header and the Contact header point to different hosts. **[1]** = Enable - the device changes the Contact header so that it points to the SAS host and therefore, the top-most Via header and the Contact header point to the same host. **Note:** Operating in this mode causes all incoming dialog requests to traverse the SAS, which may cause load problems.

Parameter	Description
Web: SAS Survivability Mode EMS: Survivability Mode CLI: sas-survivability **[SASSurvivabilityMode]**	Determines the Survivability mode used by the SAS application. **[0]** Standard = Incoming INVITE and REGISTER requests are forwarded to the defined Proxy list of SASProxySet in Normal mode and handled by the SAS application in Emergency mode (default). **[1]** Always Emergency = The SAS application does not use Keep-Alive messages towards the SASProxySet, instead it always operates in Emergency mode (as if no Proxy in the SASProxySet is available). **[2]** Ignore Register = Use regular SAS Normal/Emergency logic (same as option **[0]),** but when in Normal mode incoming REGISTER requests are ignored. **[3]** Auto-answer REGISTER = When in Normal mode, the device responds to received REGISTER requests by sending a SIP 200 OK (instead of relaying the registration requests to a Proxy), and enters the registrations in its SAS database. **[4]** Use Routing Table only in Normal mode = The device uses the IP-to-IP Routing table to route IP-to-IP SAS calls only when in SAS Normal mode (and is unavailable when SAS is in Emergency mode). This allows routing of SAS IP-to-IP calls to different destinations (and not only to the SAS Proxy Set).
Web: Enable ENUM CLI: enable-enum **[SASEnableENUM]**	Enables SAS to perform ENUM (E.164 number to URI mapping) queries when receiving INVITE messages in SAS emergency mode. **[0]** Disable (default) **[1]** Enable
Web: SAS Binding Mode EMS: Binding Mode CLI: sasbindingmode **[SASBindingMode]**	Determines the SAS application database binding mode. **[0]** URI = If the incoming AoR in the INVITE requests is using a 'tel:' URI or 'user=phone' is defined, the binding is performed according to the user part of the URI only. Otherwise, the binding is according to the entire URI, i.e., User@Host (default). **[1]** User Part only = The binding is always performed according to the User Part only.
Web: SAS Emergency Numbers CLI: sas-emerg-nb **[SASEmergencyNumbers]**	Defines emergency numbers for the device's SAS application. When the device's SAS agent receives a SIP INVITE (from an IP phone) that includes one of the emergency numbers (in the SIP user part), it forwards the INVITE to a default gateway (configured by the parameter SASDefaultGatewayIP), which sends the call directly to the PSTN. This is important for routing emergency numbers such as 911 (in North America) directly to the PSTN. This is applicable to SAS operating in Normal and Emergency modes. Up to four emergency numbers can be defined, where each number can be up to four digits.
CLI: sas-emerg-prefix **[SASEmergencyPrefix]**	Defines a prefix that is added to the Request-URI user part of the INVITE message that is sent by the device's SAS agent when in Emergency mode to the default gateway or to any other destination (using the IP2IP Routing table). This parameter is required to differentiate between normal SAS calls routed to the default gateway and emergency SAS calls. Therefore, this allows you to define different manipulation rules for normal and emergency calls. This valid value is a character string. The default is an empty string "".
Web: SAS Inbound Manipulation Mode	Enables destination number manipulation in incoming INVITE messages when SAS is in Emergency the state. The manipulation

Parameter	Description
CLI: sas-inb-manipul-md **[SASInboundManipulationMode]**	rule is done in the IP to IP Inbound Manipulation table. **[0]** None (default) **[1]** Emergency only **Notes:** Inbound manipulation applies only to INVITE requests. For more information on SAS inbound manipulation, see 'Manipulating Destination Number of Incoming INVITE' on page 217.

Parameter	Description
SAS Registration Manipulation Table	
Web: SAS Registration Manipulation EMS: Stand-Alone Survivability CLI: config voip > sas SASRegistrationManipulation **[SASRegistrationManipulation]**	This *para*meter table configures the SAS Registration Manipulation table. This table is used by the SAS application to manipulate the SIP Request-URI user part of incoming INVITE messages and of incoming REGISTER request AoR (To header), before saving it to the registered users database. The format of this table parameter is as follows: [SASRegistrationManipulation] FORMAT SASRegistrationManipulation_Index = SASRegistrationManipulation_RemoveFromRight, SASRegistrationManipulation_LeaveFromRight; [\SASRegistrationManipulation] RemoveFromRight = number of digits removed from the right side of the user part before saving to the registered user database. LeaveFromRight = number of digits to keep from the right side. If both RemoveFromRight and LeaveFromRight are defined, the RemoveFromRight is applied first. The registered database contains the AoR before and after manipulation. The range of both RemoveFromRight and LeaveFromRight is 0 to 30. For example, the manipulation rule below routes an INVITE with Request-URI header "sip:7184002@10.33.4.226" to user "4002@10.33.4.226" (i.e., keep only four digits from right of user part): `SASRegistrationManipulation 0 = 0, 4;` **Notes:** You can only configure one index entry. For a detailed description of the individual parameters in this table and for configuring this table using the Web interface, see 'Manipulating Destination Number of Incoming INVITE' on page 217.
Web: SAS IP-to-IP Routing Table	
[IP2IPRouting]	This *para*meter table configures the IP-to-IP Routing table for SAS routing rules. The format of this parameter is as follows: [IP2IPRouting] FORMAT IP2IPRouting_Index = IP2IPRouting_SrcIPGroupID, IP2IPRouting_SrcUsernamePrefix, IP2IPRouting_SrcHost, IP2IPRouting_DestUsernamePrefix, IP2IPRouting_DestHost, IP2IPRouting_DestType, IP2IPRouting_DestIPGroupID, IP2IPRouting_DestSRDID, IP2IPRouting_DestAddress, IP2IPRouting_DestPort, IP2IPRouting_DestTransportType, IP2IPRouting_AltRouteOptions;[\IP2IPRouting] For example: IP2IPRouting 1 = -1, *, *, *, *, 0, -1, -1, , 0, -1, 0; **Notes:** This table can include up to 120 indices (where 0 is the first index). For a description on configuring *ini* file table parameters, see 'Configuring ini File Table Parameters' on page 54.

Auxiliary and Configuration Files Parameters

This subsection describes the device's auxiliary and configuration files parameters.

Auxiliary/Configuration File Name Parameters

The configuration files (i.e., auxiliary files) can be loaded to the deviceusing the Web interface (see 'Loading Auxiliary Files' on page 247). For loading these files using the *ini* file, you need to configure these files in the *ini* file and configured whether they must be stored in the non-volatile memory. The table below lists the *ini* file parameters associated with these auxiliary files. For more information on the auxiliary files, see 'Auxiliary Configuration Files' on page 249.

Table A-32: Auxiliary and Configuration File Parameters

Parameter	Description
General Parameters	
[SetDefaultOnIniFileProcess]	Determines if all the device's parameters are set to their defaults before processing the updated *ini* file. **[0]** Disable - parameters not included in the downloaded *ini* file are not returned to default settings (i.e., retain their current settings). **[1]** Enable (default) **Note:** This parameter is applicable only for automatic HTTP update or Web *ini* file upload (not applicable if the *ini* file is loaded using BootP).
[SaveConfiguration]	Determines if the device's configuration (parameters and files) is saved to flash (non-volatile memory). **[0]** = Configuration isn't saved to flash memory. **[1]** = Configuration is saved to flash memory (default).
Auxiliary and Configuration File Name Parameters	
Web: Dial Plan File EMS: Dial Plan File Name **[DialPlanFileName]**	Defines the name (and path) of the Dial Plan file (defining dial plans). This file should be constructed using the DConvert utility (refer to the Product Reference Manual).
[UserInfoFileName]	Defines the name (and path) of the file containing the User Information data.

Automatic Update Parameters

The automatic update of software and configuration files parameters are described in the table below.

Table A-33: Automatic Update of Software and Configuration Files Parameters

Parameter	Description
General Automatic Update Parameters	
[AutoUpdateCmpFile]	Enables the Automatic Update mechanism for the cmp file. **[0]** = The Automatic Update mechanism doesn't apply to the cmp file (default). **[1]** = The Automatic Update mechanism includes the cmp file. **Note:** For this parameter to take effect, a device reset is required.
[AutoUpdateFrequency]	Defines the number of minutes that the device waits between automatic updates. The default value is 0 (i.e., the update at fixed intervals mechanism is disabled). **Note:** For this parameter to take effect, a device reset is required.
[AutoUpdatePredefinedTime]	Defines schedules (time of day) for automatic updates. The format of this parameter is: 'HH:MM', where *HH* denotes the hour and *MM* the minutes, for example, 20:18. **Notes:** For this parameter to take effect, a device reset is required. The actual update time is randomized by five minutes to reduce the load on the Web servers.
EMS: AUPD Verify Certificates **[AUPDVerifyCertificates]**	Determines whether the Automatic Update mechanism verifies server certificates when using HTTPS. **[0]** = Disable (default) **[1]** = Enable
[AUPDCheckIfIniChanged]	Determines whether the Automatic Update mechanism performs CRC checking to determine if the *ini* file has changed prior to processing. **[0]** = Do not check CRC. The *ini* file is loaded whenever the server provides it. (default) **[1]** = Check CRC for the entire file. Any change, including line order, causes the *ini* file to be re-processed. **[2]** = Check CRC for individual lines. Use this option when the HTTP server scrambles the order of lines in the provided *ini* file.
[ResetNow]	Invokes an immediate device reset. This option can be used to activate offline (i.e., not on-the-fly) parameters that are loaded using the parameter IniFileUrl. **[0]** = The immediate restart mechanism is disabled (default). **[1]** = The device immediately resets after an *ini* file with this parameter set to 1 is loaded.

Parameter	Description
Software/Configuration File URL Path for Automatic Update Parameters	
[CmpFileURL]	Defines the name of the *cmp* file and the path to the server (IP address or FQDN) from where the device can load the *cmp* file and update itself. The *cmp* file can be loaded using HTTP/HTTPS. For example: http://192.168.0.1/filename **Notes:** For this parameter to take effect, a device reset is required. When this parameter is configured, the device always loads the *cmp* file after it is reset. The *cmp* file is validated before it's burned to flash. The checksum of the *cmp* file is also compared to the previously burnt checksum to avoid unnecessary resets. The maximum length of the URL address is 255 characters.
[IniFileURL]	Defines the name of the *ini* file and the path to the server (IP address or FQDN) on which it is located. The *ini* file can be loaded using HTTP/HTTPS. For example: http://192.168.0.1/filename http://192.8.77.13/config<MAC> https://<username>:<password>@<IP address>/<file name> **Notes:** For this parameter to take effect, a device reset is required. When using HTTP or HTTPS, the date and time of the *ini* file are validated. Only more recently dated *ini* files are loaded. The optional string <MAC>' is replaced with the device's MAC address. Therefore, the device requests an *ini* file name that contains its MAC address. This option allows the loading of specific configurations for specific devices. The maximum length of the URL address is 99 characters.
[TLSRootFileUrl]	Defines the name of the TLS trusted root certificate file and the URL from where it can be downloaded. **Note:** For this parameter to take effect, a device reset is required.
[TLSCertFileUrl]	Defines the name of the TLS certificate file and the URL from where it can be downloaded. **Note:** For this parameter to take effect, a device reset is required.
[TLSPkeyFileUrl]	Defines the URL for downloading a TLS private key file using the Automatic Update facility.
[UserInfoFileURL]	Defines the name of the User Information file and the path to the server (IP address or FQDN) on which it is located. For example: http://server_name/file, https://server_name/file **Note:** The maximum length of the URL address is 99 characters.

SIP Message Manipulation Syntax

This section provides a detailed description on the support and syntax for configuring SIP message manipulation rules. For configuring message manipulation rules, see 'Configuring Message Manipulations' on page 192.

Actions

The actions that can be done on SIP message manipulation in the Message Manipulations table are listed in the table below.

Table B-1: Message Manipulation Actions

Action	Value
Add	0
Remove	1
Modify	2
Add Prefix	3
Add Suffix	4
Remove Suffix	5
Remove Prefix	6

The maximum length of the value for a manipulation is 299 characters.

Header Types

Accept

An example of the header is shown below:

```
Accept:  application/sdp
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	No	N/A
Keyword	**Sub Types**			**Attributes**
N/A	N/A			N/A

Below is a header manipulation example:

Rule:	If the supported header does not contain 'mm,100rel,timer,replaces', then in all INVITE messages add an Accept header: `MessageManipulations 8 = 1, invite, header.supported != 'mm,100rel,timer,replaces', header.accept, 0, ' application/x-private ', 0;`
Result:	`Accept: application/x-private`

Accept-Language

An example of the header is shown below:

```
Accept-Language: da, en-gb;q=0.8, en;q=0.7
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	No	N/A
Keyword	**Sub Types**		**Attributes**	
N/A	N/A		N/A	

Below is a header manipulation example:

Rule:	Add a new Language header to all INVITE messages: `MessageManipulations 0 = 1, invite, , header.accept-language, 0, 'en, il, cz, it', 0;`
Result:	`Accept-Language: en, il, cz, it`

Allow

An example of the header is shown below:

```
Allow:
REGISTER,OPTIONS,INVITE,ACK,CANCEL,BYE,NOTIFY,PRACK,REFER,INFO,SUBSCRIBE
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	No	N/A

Keyword	Sub Types	Attributes
N/A	N/A	Read/Write

Below is a header manipulation example:

Rule:	Add an Allow header to all INVITE messages: `MessageManipulations 0 = 1, invite, , header.allow, 0, 'REGISTER,OPTIONS,INVITE,ACK,CANCEL,BYE,NOTIFY,PRACK,REFER,INFO,SUBSCRIBE, XMESSAGE', 0;`
Result:	`Allow: REGISTER,OPTIONS,INVITE,ACK,CANCEL,BYE,NOTIFY,PRACK,REFER,INFO,SUBSCRIBE, XMESSAGE`

Call-Id

An example of the header is shown below:

```
Call-ID: JNIYXOLCAIWTRHWOINNR@10.132.10.128
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	No	No	No	NA

Keyword	Sub Types	Attributes
ID	String	Read Only

Below is a header manipulation example:

Rule:	Add a proprietary header to all INVITE messages using the data in the Call-id header: `MessageManipulations 0 = 1, invite, , header.Xitsp-abc, 0, header.call-id, 0;`
Result:	`Xitsp-abc: GIAPOFWRBQKJVAETIODI@10.132.10.128`

Contact

An example of the header is shown below:

```
Contact: <sip:555@10.132.10.128:5080>
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	No	No	No	8

Keyword	Sub Types	Attributes
Expires	Integer	Read/Write
GruuContact	String	Read/Write
IsGRUU	Boolean	Read/Write
Name	String	Read/Write
Param	Param	Read/Write
URL	'URL' on page 391	Read/Write*

* Host name cannot be modified in the URL structure for a contact header.

Below is a header manipulation example:

Rule:	Change the user part in the Contact header in all INVITE messages to fred: `MessageManipulations 0 = 1, Invite, ,header.contact.url.user, 2, 'fred', 0;`
Result:	`Contact: <sip:fred@10.132.10.128:5070>`

Cseq

An example of the header is shown below:

```
CSeq: 1 INVITE
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	No	No	No	N/A

Keyword	Sub Types	Attributes
Num	Integer	Read Only
Type	String	Read Only

Below is a header manipulation example:

Rule:	If the Cseq number is 1, then modify the user in the Contact header to fred. `MessageManipulations 0 = 1, Invite, header.cseq.num=='1',header.contact.url.user, 2, 'fred', 0;`
Result:	`Contact: <sip:fred@10.132.10.128:5070>`

Diversion

An example of the header is shown below:

```
Diversion: <sip:654@IPG2Host;user=phone>;reason=user-
busy;screen=no;privacy=off;counter=1
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	3

Keyword	Sub Types	Attributes
Name	String	Read/Write
Param	Param	Read/Write
Privacy	Enum Privacy (see 'Privacy' on page 398)	Read/Write
Reason	Enum Reason (see 'Reason (Diversion)' on page 398)	Read/Write
Screen	Enum Screen (see 'Screen' on page 401)	Read/Write
URL	URL Structure (see 'URL' on page 391)	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add a Diversion header to all INVITE messages: `MessageManipulations 0 = 1, invite, , header.Diversion, 0, '<tel:+101>;reason=unknown; counter=1;screen=no; privacy=off', 0;`
	Result:	`Diversion: <tel:+101>;reason=user-busy;screen=no;privacy=off;counter=1`
Example 2	**Rule:**	Modify the Reason parameter in the header to 1, see 'Reason (Diversion)' on page 398for possible values: `MessageManipulations 1 = 1, invite, , header.Diversion.reason, 2, '1', 0;`
	Result:	`Diversion: <tel:+101>;reason=user-busy;screen=no;privacy=off;counter=1`
Example 3	**Rule:**	The URL in the Diversion header is modified to that which is contained in the header URL: `MessageManipulations 2 = 1, invite, , header.Diversion.URL, 2, header.from.url, 0;`
	Result:	`Diversion:<sip:555@IPG2Host;user=phone>;reason=user-busy;screen=no;privacy=off;counter=1`

Event

An example of the header is shown below:

```
Event: foo; id=1234
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	N/A

Keyword	Sub Types	Attributes
EventKey	Event Structure (see 'Event Structure' on page 389)	Read/Write
Param	Param	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add parameter itsp-abc=voip to the Event header: `MessageManipulations 0 = 1, invite, , header.event.param.itsp-abc, 0, 'voip' , 0;`
	Result:	`Event: foo;id=1234;itsp-abc=voip`
Example 2	**Rule:**	Modify the Event ID string: `MessageManipulations 1 = 1, invite, , header.event.EVENTKEY.id, 2, '5678', 0;`
	Result:	`Event: foo;id=5678;`
Example 3	**Rule:**	Modify the Event package enum: `MessageManipulations 2 = 1, invite, , header.event.EVENTKEY.EVENTPACKAGE, 2, '2', 0;`
	Result:	`Event: refer;id=5678`

From

An example of the header is shown below:

```
From: <sip:555@10.132.10.128;user=phone>;tag=YQLQHCAAYBWKKRVIMWEQ
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	No	No	No	NA

Keyword	Sub Types	Attributes
Name	String	Read/Write
Param	Param	Read/Write
tag	String	Read Only
URL	URL Structure (refer to 'URL' on page 391)	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Change the user part of the From header if the user is not 654: `MessageManipulations 8 = 1, invite, header.from.url.user != '654', header.from.url.user, 2, 'fred', 0;`
	Result:	`From: <sip:fred@IPG2Host;user=phone>;tag=1c20161`
Example 2	**Rule:**	Add a new parameter to the From header called p1 and set its value to myParameter: `MessageManipulations 1 = 1, Invite.request, ,header.from.param.p1, 0, 'myParameter', 0;`
	Result:	`From: <sip:fred@IPG2Host;user=phone>;p1=myParameter;tag=1c5891`
Example 3	**Rule:**	Modify the URL in the From header: `MessageManipulations 0 = 1, any, , header.from.url, 2, 'sip:3200@110.18.5.41;tusunami=0', 0;`
	Result:	`From: <sip:3200@110.18.5.41;user=phone;tusunami=0>;tag=1c23750`

History-Info

An example of the header is shown below:

```
History-Info: <sip:UserA@ims.example.com;index=1>
History-Info: <sip:UserA@audc.example.com;index=2>
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	20

Keyword	Sub Types	Attributes
HistoryInfo	String	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add a new History-Info header to the message: `MessageManipulations 0 = 1, any, , header.History-Info, 0, '<sip:UserA@audc.mydomain.com;index=3>', 0`

	Result:	`History-Info:sip:UserA@ims.example.com;index=1`
		`History-Info:sip:UserA@audc.example.com;index=2`
		`History-Info: <sip:UserA@audc.mydomain.com;index=3>`

Example 2	**Rule:**	Delete an unwanted History-Info header from the message: `MessageManipulations 0 = 1, any, , header.History-Info.1, 1, , 0;`
	Result:	`History-Info: <sip:UserA@ims.example.com;index=1>`
Example 3	**Rule:**	Delete all History-Info from the message: `MessageManipulations 0 = 1, any, , header.History-Info, 1, , 0;`
	Result:	`All history-info headers are removed.`

Min-Se and Min-Expires

An example of the header is shown below:

```
Min-SE: 3600
Min-Expires: 60
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	N/A

Keyword	Sub Types	Attributes
Param	Param	Read/Write
Time	Integer	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add a Min-Se header to the message using a value of 50: `MessageManipulations 1 = 1, any, , header.min-se, 0, '50', 0;`
	Result:	`Min-SE: 50`
Example 2	**Rule:**	Modify a Min-Expires header with the min-expires value and add an additional 0: `MessageManipulations 0 = 1, Invite, , header.Min-Expires.param, 2, header.Min-Expires.time + '0', 0;`
	Result:	`Min-Expires: 340;3400`
Example 3	**Rule:**	Modify a Min-Expires header changing the time to 700: `MessageManipulations 0 = 1, Invite, , header.Min-Expires.time, 2, '700', 0;`
	Result:	`Min-Expires: 700`

P-Asserted-Identity

An example of the header is shown below:

```
P-Asserted-Identity: Jane Doe <sip:567@itsp.com>
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	1

Keyword	Sub Types	Attributes
URL	URL Structure (see 'URL' on page 391)	Read/Write
Name	String	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add a P-Asserted-Id header to all INVITE messages: `MessageManipulations 2 = 1, invite, , header.p-asserted-identity, 0, '<sip:567@itsp.com>', 0;`
	Result:	`P-Asserted-Identity: <sip:567@itsp.com>`
Example 2	**Rule:**	Modify the P-Asserted-Identity host name to be the same as the host name in the To header: `MessageManipulations 2 = 1, invite, , header.p-asserted-identity.URL.host, 2, header.to.url.host, 0;`
	Result:	`P-Asserted-Identity: <sip:567@10.132.10.128>`

P-Associated-Uri

An example of the header is shown below:

```
P-Associated-URI: <sip:12345678@itsp.com>
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	1

Keyword	Sub Types	Attributes
Name	String	Read/Write
Param	Param	Read/Write
URL	URL Structure (see 'URL' on page 391)	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add a P-Associated-Uri header to all INVITE response messages: `MessageManipulations 5 = 1, register.response, ,header.P-Associated-URI, 0, '<sip:admin@10.132.10.108>', 0;`
	Result:	`P-Associated-URI:<sip:admin@10.132.10.108>`
Example 2	**Rule:**	Modify the user portion of the URL in the header to 'alice': `MessageManipulations 5 = 1, register.response, ,header.P-Associated-URI.url.user, 2, 'alice', 0;`
	Result:	`P-Associated-URI:<sip:alice@10.132.10.108>`

P-Called-Party-Id

An example of the header is shown below:

```
P-Called-Party-ID: <sip:2000@gw.itsp.com>
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	N/A

Keyword	Sub Types	Attributes
Name	String	Read/Write
URL	URL Structure (see 'URL' on page 391)	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add a P-Called-Party-Id header to all messages: `MessageManipulations 8 = 1, any, , header.p-called-party-id, 0, 'sip:2000@MSBG.ITSP.COM', 0;`

	Result:	`P-Called-Party-ID: <sip:2000@gw.itsp.com>`
Example 2	**Rule:**	Append a parameter (p1) to all P-Called-Party-Id headers: `MessageManipulations 9 = 1, invite, , header.p-called-party-id.param.p1, 0, 'red', 0;`
	Result:	`P-Called-Party-ID: <sip:2000@gw.itsp.com>;p1=red`
Example 3	**Rule:**	Add a display name to the P-Called-Party-Id header: `MessageManipulations 3 = 1, any, , header.p-called-party-id.name, 2, 'Secretary', 0;`
	Result:	`P-Called-Party-ID: Secretary <sip:2000@gw.itsp.com>;p1=red`

P-Charging-Vector

An example of the header is shown below:

```
P-Charging-Vector: icid-value=1234bc9876e; icid-generated-at=192.0.6.8; orig-ioi=home1.net
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	No	N/A

Keyword	Sub Types	Attributes
N/A	N/A	N/A

Below are header manipulation examples:

Rule:	Add a P-Charging-Vector header to all messages: `MessageManipulations 1 = 1, any, , header.P-Charging-Vector, 0, 'icid-value=1234bc9876e; icid-generated-at=192.0.6.8; orig-ioi=home1.net', 0;`
Result:	`P-Charging-Vector: icid-value=1234bc9876e; icid-generated-at=192.0.6.8; orig-ioi=home1.net`

P-Preferred-Identity

An example of the header is shown below:

```
P-Preferred-Identity: "Cullen Jennings" <sip:fluffy@abc.com>
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	N/A

Keyword	Sub Types	Attributes
Name	String	Read/Write
URL	URL Structure (see 'URL' on page 391)	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add a P-Preferred-Identity header to all messages: `MessageManipulations 1 = 1, any, , header.P-Preferred-Identity, 0, 'Cullen Jennings <sip:fluffy@abc.com>', 0;`
	Result:	`P-Preferred-Identity: "Cullen Jennings" <sip:fluffy@abc.com>`
Example 2	**Rule:**	Modify the display name in the P-Preferred-Identity header: `MessageManipulations 2 = 1, any, , header.P-Preferred-Identity.name, 2, 'Alice Biloxi', 0;`
	Result:	`P-Preferred-Identity: "Alice Biloxi" <sip:fluffy@abc.com>`

Privacy

An example of the header is shown below:

```
Privacy: none
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	No	N/A

Keyword	Sub Types	Attributes
privacy	'Privacy Struct' on page 390	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add a privacy header and set it to ‘‘session': `MessageManipulations 1 = 1, any, , header.Privacy, 0, 'session', 0;`
	Result:	`Privacy: session`
Example 2	**Rule:**	Add ‘user’ to the list: `MessageManipulations 1 = 3, , ,`

Example 1	**Rule:**	Add a privacy header and set it to ‘‘session': `MessageManipulations 1 = 1, any, , header.Privacy, 0, 'session', 0;`
	Result:	`Privacy: session`
		`header.privacy.privacy.user, 2, '1', 0;`
	Result:	`Privacy: session;user`

Proxy-Require

An example of the header is shown below:

```
Proxy-Require: sec-agree
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	N/A

Keyword	Sub Types	Attributes
Capabilities	SIPCapabilities Struct	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add a Proxy-Require header to the message: `MessageManipulations 1 = 1, any, , header.Proxy-Require, 0, 'sec-agree', 0;`
	Result:	`Proxy-Require: sec-agree`
Example 2	**Rule:**	Modify the Proxy-Require header to itsp.com: `MessageManipulations 2 = 1, any, , header.Proxy-Require, 2, 'itsp.com' , 0;`
	Result:	`Proxy-Require: itsp.com`
Example 3	**Rule:**	Set the privacy options tag in the Proxy-Require header: `MessageManipulations 0 = 0, invite, , header. Proxy-Require.privacy, 0, 1 , 0;`
	Result:	`Proxy-Require: itsp.com, privacy`

Reason

An example of the header is shown below:

```
Reason: SIP ;cause=200 ;text="Call completed elsewhere"
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	N/A

Keyword	Sub Types	Attributes
MLPP	MLPP Structure (see 'MLPP' on page 389)	Read/Write
Reason	Reason Structure (see 'Reason Structure' on page 390)	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add a Reason header: `MessageManipulations 0 = 1, any, ,header.reason, 0, 'SIP;cause=200;text="Call completed elsewhere"', 0;`
	Result:	`Reason: SIP ;cause=200 ;text="Call completed elsewhere"`
Example 2	**Rule:**	Modify the reason cause number: `MessageManipulations 0 = 1, any, ,header.reason.reason.cause, 0, '200', 0;`
	Result:	`Reason: Q.850 ;cause=180 ;text="Call completed elsewhere"`
Example 3	**Rule:**	Modify the cause number: `MessageManipulations 0 = 1, any, ,header.reason.reason.reason, 0, '483', 0;`
	Result:	`Reason: SIP ;cause=483 ;text="483 Too Many Hops"`

Note: The protocol (SIP or Q.850) is controlled by setting the cause number to be greater than 0. If the cause is 0, then the text string (see Example 3) is generated from the reason number.

Referred-By

An example of the header is shown below:

```
Referred-By: <sip:referrer@referrer.example>;
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	N/A

Keyword	Sub Types	Attributes
param	param	Read/Write
URL	URL Structure (see 'URL' on page 391)	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add a Referred-By header: `MessageManipulations 0 = 1, any, ,header.Referred-By, 0, '<sip:refer@refer.com>', 0;`
	Result:	`Referred-By: <sip: sip:refer@refer.com>`
Example 2	**Rule:**	Modify the host: `MessageManipulations 0 = 1, any, ,header.Referred-By.url.host, 0, 'yahoo.com', 0;`
	Result:	`Referred-By: <sip:refer@yahoo.com>`
Example 3	**Rule:**	Add a new parameter to the header: `MessageManipulations 0 = 1, any, ,header.Referred-By.param.p1, 0, 'fxs', 0`
	Result:	`Referred-By: <sip:referrer@yahoo.com>;p1=fxs`

Refer-To

An example of the header is shown below:

```
Refer-To: sip:conference1@example.com
Refer-To:
<sips:a8342043f@atlanta.example.com?Replaces=12345601%40atlanta.examp
le.com%3bfrom-tag%3d314159%3bto-tag%3d1234567>
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	No	N/A

Keyword	Sub Types	Attributes
N/A	N/A	N/A

Below are header manipulation examples:

Example 1	**Rule:**	Add a basic header: `MessageManipulations 0 = 1, any, ,header.Refer-to, 0, '<sip:referto@referto.com>', 0;`
	Result:	`Refer-To: <sip:referto@referto.com>`
Example 2	**Rule:**	Add a Refer-To header with URI headers: `MessageManipulations 0 = 1, any, ,header.Refer-to, 0, '<sips:a8342043f@atlanta.example.com?Replaces=12345601%40 atlanta.example.com%3bfrom-tag%3d314159%3bto- tag%3d1234567>', 0;`
	Result:	`Refer-To: <sips:a8342043f@atlanta.example.com?Replaces=12345601%40a tlanta.example.com%3bfrom-tag%3d314159%3bto- tag%3d1234567>`

Remote-Party-Id

An example of the header is shown below:

```
Remote-Party-ID: "John Smith"
<sip:john.smith@itsp.com>;party=calling; privacy=full;screen=yes
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	3

Keyword	Sub Types	Attributes
Counter	Integer	Read/Write
Name	String	Read/Write
NumberPlan	Enum Number Plan (see 'Number Plan' on page 397)	Read/Write
NumberType	Enum Number Type (see 'NumberType' on page 397)	Read/Write
Param	Param	Read/Write
Privacy	Enum Privacy (see 'Privacy' on page 398)	Read/Write
Reason	Enum Reason (RPI) (see 'Reason (Remote-Party-Id)' on page 401)	Read/Write
Screen	Enum Screen (see 'Screen' on page 401)	Read/Write
ScreenInd	Enum ScreenInd (see 'ScreenInd' on page 401)	Read/Write
URL	URL Structure (see 'URL' on page 391)	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add a Remote-Party-Id header to the message: `MessageManipulations 0 = 1, invite, ,header.REMOTE-PARTY-ID, 0, '<sip:999@10.132.10.108>;party=calling', 0;`
	Result:	`Remote-Party-ID: <sip:999@10.132.10.108>;party=calling;npi=0;ton=0`
Example 2	**Rule:**	Create a Remote-Party-Id header using the url in the From header using the + operator to concatenate strings: `MessageManipulations 0 = 1, Invite, ,header.REMOTE-PARTY-ID, 0, '<'+header.from.url +'>' + ';party=calling', 0;`
	Result:	`Remote-Party-ID: <sip:555@10.132.10.128;user=phone>;party=calling;npi=0;ton=0`
Example 3	**Rule:**	Modify the number plan to 1 (ISDN): `MessageManipulations 1 = 1, invite, , header.Remote-Party-ID.numberplan, 2, '1', 0;`
	Result:	`Remote-Party-ID: <sip:555@10.132.10.128;user=phone>;party=calling;npi=1;ton=0`
Example 4	**Rule:**	Modify the Remote-Party-Id header to set the privacy parameter to 1 (Full): `MessageManipulations 1 = 1, invite, , header.Remote-Party-ID.privacy, 2, '1', 0;`

	Result:	`Remote-Party-ID: <sip:555@10.132.10.128;user=phone>;party=calling;privacy=full;npi=0;ton=0`

Request-Uri

An example of the header is shown below:

```
sip:alice:secretword@atlanta.com;transport=tcp
SIP/2.0 486 Busy Here
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	No	No	Yes	NA

Keyword	Sub Types	Attributes
Method	String	Read/Write
MethodType	Enum	Read/Write
URI	String	Read/Write
URL	URL Structure (see 'URL' on page 391)	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Test the Request-URI transport type. If 1 (TCP), then modify the URL portion of the From header: `MessageManipulations 1 = 1, Invite.request, header.REQUEST-URI.url.user == '101', header.REMOTE-PARTY-ID.url, 2, 'sip:3200@110.18.5.41;tusunami=0', 0;`
	Result:	`Remote-Party-ID: <sip:3200@110.18.5.41;tusunami=0>;party=calling;npi=0;ton=0`
Example 2	**Rule:**	If the method type is 5 (INVITE), then modify the Remote-Party-Id header: `MessageManipulations 2 = 1, Invite.request, header.REQUEST-URI.methodtype == '5', header.REMOTE-PARTY-ID.url, 2, 'sip:3200@110.18.5.41;tusunami=0', 0;`
	Result:	`Remote-Party-ID: <sip:3200@110.18.5.41;tusunami=0>;party=calling;npi=0;ton=0`
Example 3	**Rule:**	For all request URI's whose method types are 488, modify the message type to a 486: `MessageManipulations 1 = 1, , header.request-uri.methodtype=='488', header.request-uri.methodtype, 2, '486', 0;`
	Result:	`SIP/2.0 486 Busy Here`

Require

An example of the header is shown below:

```
Require: 100rel
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	N/A

Keyword	Sub Types	Attributes
Capabilities	SIPCapabilities Struct	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add a Require header to all messages: `MessageManipulations 1 = 1, , ,header.require, 0, 'early-session,em,replaces', 0;`
	Result:	`Require: em,replaces,early-session`
Example 2	**Rule:**	If a Require header exists, then delete it: `MessageManipulations 2 = 1, Invite, header.require exists ,header.require, 1, '', 0;`
	Result:	The Require header is deleted.
Example 3	**Rule:**	Set the early media options tag in the header: `MessageManipulations 0 = 0, invite, , header.require.earlymedia, 0, 1 , 0;`
	Result:	`Require: em,replaces,early-session, early-media`
Example 4	**Rule:**	Set the privacy options tag in the Require header: `MessageManipulations 0 = 0, invite, , header.require.privacy, 0, 1 , 0;`
	Result:	`Require: em,replaces,early-session, privacy`

Resource-Priority

An example of the header is shown below:

```
Resource-Priority: wps.3
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	2

Keyword	Sub Types	Attributes
Namespace	String	Read/Write
RPriority	String	Read/Write

Retry-After

An example of the header is shown below:

```
Retry-After: 18000
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	N/A

Keyword	Sub Types	Attributes
Time	Integer	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add a Retry-After header: `MessageManipulations 2 = 1, Invite,  ,header.Retry-After, 0, '3600', 0;`
	Result:	`Retry-After: 3600`
Example 2	**Rule:**	Modify the Retry-Time in the header to 1800: `MessageManipulations 3 = 1, Invite,  ,header.Retry-After.time, 2, '1800', 0;`
	Result:	`Retry-After: 1800`

Server or User-Agent

An example of the header is shown below:

```
User-Agent: Sip Message Generator V1.0.0.5
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	N/A

Keyword	Sub Types	Attributes
N/A	N/A	N/A

Below are header manipulation examples:

Example 1	**Rule:**	Remove the User-Agent header: `MessageManipulations 2 = 1, Invite,  ,header.user-agent, 1, '', 0;`
	Result:	The header is removed.
Example 2	**Rule:**	Change the user agent name in the header: `MessageManipulations 3 = 1, Invite,  ,header.user-agent, 2, 'itsp analogue gateway', 0;`
	Result:	`User-Agent: itsp analog gateway`

Service-Route

An example of the header is shown below:

```
Service-Route: <sip:P2.HOME.EXAMPLE.COM;lr>,
<sip:HSP.HOME.EXAMPLE.COM;lr>
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	7

Keyword	Sub Types	Attributes
ServiceRoute	String	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add two Service-Route headers:
		`MessageManipulations 1 = 1, Invite, ,header.service-route, 0, '<P2.HOME.EXAMPLE.COM;lr>', 0;`
		`MessageManipulations 2 = 1, Invite, ,header.service-route, 0, '<sip:HSP.HOME.EXAMPLE.COM;lr>', 0;`

	Result:	`Service-Route:<P2.HOME.EXAMPLE.COM;lr>` `Service-Route: <sip:HSP.HOME.EXAMPLE.COM;lr>`
Example 2	**Rule:**	Modify the Service-Route header in list entry 1: `MessageManipulations 3 = 1, Invite, ,header.service-route.1.serviceroute, 2, '<sip:itsp.com;lr>', 0;`
	Result:	`Service-Route:sip:itsp.com;lr` `Service-Route: <sip:HSP.HOME.EXAMPLE.COM;lr>`
Example 3	**Rule:**	Modify the Service-Route header in list entry 0: `MessageManipulations 4 = 1, Invite, ,header.service-route.0.serviceroute, 2, '<sip:home.itsp.com;lr>', 0;`
	Result:	`Service-Route:sip:home.itsp.com;lr` `Service-Route: <sip:itsp.com;lr>`

Session-Expires

An example of the header is shown below:

```
Session-Expires: 480
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	N/A

Keyword	Sub Types	Attributes
Param	Param	Read/Write
Refresher	Enum Refresher (see 'Refresher' on page 401)	Read/Write
Time	Integer	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add a Session-Expires header: `MessageManipulations 0 = 1, any, , header.Session-Expires, 0, '48' + '0', 0;`
	Result:	`Session-Expires: 480`
Example 2	**Rule:**	Modify the Session-Expires header to 300: `MessageManipulations 1 = 1, any, , header.Session-Expires.time, 2, '300', 0;`
	Result:	`Session-Expires: 300`
Example 3	**Rule:**	Add a param called longtimer to the header: `MessageManipulations 1 = 1, any, , header.Session-Expires.param.longtimer, 0, '5', 0;`
	Result:	`Session-Expires: 480;longtimer=5`
Example 4	**Rule:**	Set the refresher to 1 (UAC): `MessageManipulations 3 = 1, any, , header.session-expires.refresher, 2, '1', 0;`
	Result:	`Session-Expires: 300;refresher=uac;longtimer=5`

Subject

An example of the header is shown below:

```
Subject: A tornado is heading our way!
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	N/A

Keyword	Sub Types	Attributes
Subject	String	Read/Write

Below is a header manipulation example:

Rule:	Add a Subject header: `MessageManipulations 0 = 1, any, , header.Subject, 0, 'A tornado is heading our way!', 0;`
Result:	`Subject: A tornado is heading our way!`

Supported

An example of the header is shown below:

```
Supported: early-session
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	N/A

Keyword	Sub Types	Attributes
Capabilities	SIPCapabilities Struct	Read/Write

Below is a header manipulation example:

Example 1	**Rule:**	Add a Supported header: `MessageManipulations 1 = 1, Invite, ,header.supported, 0, 'early-session, 0;`
	Result:	`Supported: early-session`
Example 2	**Rule:**	Set path in the Supported headers options tag: `MessageManipulations 0 = 0, invite, , header.supported.path, 0, true, 0;`
	Result:	`Supported: early-session, path`

To

An example of the header is shown below:

```
To: <sip:101@10.132.10.128;user=phone>
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	No	No	No	NA

Keyword	Sub Types	Attributes
Name	String	Read/Write
Param	Param	Read/Write
tag	String	Read Only
URL	URL Structure (refer to 'URL' on page 391)	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Set the user phone Boolean to be false in the To header's URL: `MessageManipulations 4 = 1, invite.request, , header.to.url.UserPhone, 2, '0', 0;`
	Result:	`To: <sip:101@10.132.10.128>`
Example 2	**Rule:**	Change the URL in the To header: `MessageManipulations 4 = 1, invite.request, , header.to.url.UserPhone, 2, '0', 0;`
	Result:	`To: <sip:101@10.20.30.60:65100>`
Example 3	**Rule:**	Set the display name to 'Bob': `MessageManipulations 5 = 1, invite.request, , header.to.name, 2, 'Bob', 0;`
	Result:	`To: "Bob Dylan" sip:101@10.20.30.60:65100`
Example 4	**Rule:**	Add a proprietary parameter to all To headers: `MessageManipulations 6 = 1, invite.request, , header.to.param.artist, 0, 'singer', 0;`
	Result:	`To: "Bob Dylan" <sip:101@10.20.30.60:65100>;artist=singer`

Unsupported

An example of the header is shown below:

```
Unsupported: 100rel
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	N/A

Keyword	Sub Types	Attributes
Capabilities	SIPCapabilities Struct	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add an Unsupported header to the message: `MessageManipulations 0 = 1, Invite.response, ,header.unsupported, 0, 'early-session, myUnsupportedHeader', 0;`
	Result:	`Unsupported: early-session`
Example 2	**Rule:**	Modify the Unsupported header to 'replaces': `MessageManipulations 1 = 1, Invite, ,header.unsupported, 2, 'replaces', 0;`
	Result:	`Unsupported: replaces`
Example 3	**Rule:**	Set the path in the Unsupported headers options tag: `MessageManipulations 0 = 0, invite, , header.unsupported.path, 0, true, 0;`
	Result:	`Unsupported: replaces, path`

Via

An example of the header is shown below:

```
Via: SIP/2.0/UDP 10.132.10.128;branch=z9hG4bKUGOKMQPAVFKTAVYDQPTB
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	No	No	No	10

Keyword	Sub Types	Attributes
Alias	Boolean	Read Only
Branch	String	Read Only
Host	Host Structure (see 'Host' on page 389)	Read Only
MAddrIp	gnTIPAddress	Read Only
Param	Param	Read/Write

Keyword	Sub Types	Attributes
Port	Integer	Read Only
TransportType	Enum TransportType (see 'TransportType' on page 402)	Read Only

Below is a header manipulation example:

Rule:	Check the transport type in the first Via header and if it's set to UDP, then modify the From header's URL: `MessageManipulations 0 = 1, Invite.request, header.VIA.0.transporttype == '0', header.from.url, 2, 'sip:3200@110.18.5.41;tusunami=0', 0;`
Result:	`From: <sip:3200@110.18.5.41;user=phone;tusunami=0>;tag=1c7874`

Warning

An example of the header is shown below:

```
Warning: 307 isi.edu "Session parameter 'foo' not understood"
Warning: 301 isi.edu "Incompatible network address type 'E.164'"
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	1

Keyword	Sub Types	Attributes
N/A	N/A	N/A

Below is a header manipulation example:

Rule:	Add a Warning header to the message: `MessageManipulations 0 = 1, Invite.response.180, ,header.warning, 0, 'Incompatible 380', 0;`
Result:	`Warning: Incompatible 380`

Unknown Header

An Unknown header is a SIP header that is not included in this list of supported headers. An example of the header is shown below:

```
MYEXP: scooby, doo, goo, foo
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	3

Keyword	Sub Types	Attributes
N/A	N/A	N/A

Below are header manipulation examples:

Example 1	**Rule:**	Add a custom header to all messages: `MessageManipulations 0 = 1, , , header.myExp, 0, 'scooby, doo, goo, foo', 0;`
	Result:	`MYEXP: scooby, doo, goo, foo`
Example 2	**Rule:**	Take the value from the Expires parameter in the Contact header, append 00 to the value and create a new myExp header: `MessageManipulations 0 = 1, any, , header.media, 0, header.Session-Expires.time + 'ooo' + ';refresher=' + header.Session-Expires.Refresher, 0;`
	Result:	`MEDIA: 3600ooo;refresher=1`

Example 3	**Rule:**	Create lists of Unknown headers:
		`MessageManipulations 1 = 1, Invite, , header.myExp.1, 0, 'scooby, doo, goo, foo1', 0;`
		`MessageManipulations 2 = 1, Invite, , header.myExp.2, 0, 'scooby, doo, goo, foo2', 0;`

	Result:	`MYEXP: scooby, doo, goo, foo1` `MYEXP: scooby, doo, goo, foo2`
Example 4	**Rule:**	Remove the SIP header 'colour' from INVITE messages: `MessageManipulations 1 = 1, Invite, , header.colour, 1, '', 0;`
	Result:	The colour header is removed.

Structure Definitions

Event Structure

The Event structure is used in the Event header (see 'Event' on page 366).

Table B-2: Event Structure

Keyword	Sub Types	Attributes
EventPackage	Enum Event Package (see 'Event Package' on page 396)	Read/Write
EventPackageString*	String	Read/Write
Id	String	Read/Write

Event package string is used for packages that are not listed in the Enum Event Package table (see 'Event Package' on page 396).

Host

The host structure is applicable to the URL structure (see 'URL' on page 391) and the Via header (see 'Via' on page 386).

Table B-3: Host Structure

Keyword	Sub Types
Port	Short
Name	String

MLPP

This structure is applicable to the Reason header (see 'Reason' on page 374).

Table B-4: MLPP Structure

Keyword	Sub Types
Type	Enum MLPP Reason (see 'MLPP Reason Type' on page 397)
Cause	Int

Privacy Struct

This structure is applicable to the Privacy header (see 'Privacy' on page 372).

Table B-5: Privacy Structure

Keyword	Sub Types
NONE	Boolean
HEADER	Boolean
SESSION	Boolean
USER	Boolean
CRITICAL	Boolean
IDENTITY	Boolean
HISTORY	Boolean

Reason Structure

This structure is applicable to the Reason header (see 'Reason' on page 374).

Table B-6: Reason Structure

Keyword	Sub Types
Reason	Enum Reason (see 'Reason (Reason Structure)' on page 398)
Cause	Int
Text	String

SIPCapabilities

This structure is applicable to the following headers:

Supported (see 'Supported' on page 384)

Require (see 'Require' on page 378)

Proxy-Require (see 'Proxy-Require' on page 373)

Unsupported (see 'Unsupported' on page 386)

Table B-7: SIPCapabilities Structure

Keyword	Sub Types
EarlyMedia	Boolean
ReliableResponse	Boolean
Timer	Boolean
EarlySession	Boolean
Privacy	Boolean
Replaces	Boolean
History	Boolean

Keyword	Sub Types
Unknown	Boolean
GRUU	Boolean
ResourcePriority	Boolean
TargetDialog	Boolean
SdpAnat	Boolean

URL

This structure is applicable to the following headers:
Contact (see 'Contact' on page 364)
Diversion (see 'Diversion' on page 365)
From (see 'From' on page 367)
P-Asserted-Identity (see 'P-Asserted-Identity' on page 369)
P-Associated-Uri (see 'P-Associated-Uri' on page 370)
P-Called-Party-Id (see 'P-Called-Party-Id' on page 370)
P-Preferred-Identity (see 'P-Preferred-Identity' on page 372)
Referred-By (see 'Referred-By' on page 375)
Refer-To (see 'Refer-To' on page 376)
Remote-Party-Id (see 'Remote-Party-Id' on page 377)
Request-Uri (see 'Request-Uri' on page 378)
To (see 'To' on page 385)

Table B-8: URL Structure

Keyword	Sub Types
Type	Enum Type (see 'Type' on page 402)
Host	Host Structure (see 'Host' on page 389)
MHost	Structure
UserPhone	Boolean
LooseRoute	Boolean
User	String
TransportType	Enum Transport (see 'TransportType' on page 402)
Param	Param

Random Type

Manipulation rules can include random strings and integers. An example of a manipulation rule using random values is shown below:

```
MessageManipulations 4 = 1, Invite.Request, , Header.john, 0,
rand.string.56.A.Z, 0;
```

In this example, a header called "john" is added to all INVITE messages received by the device and a random string of 56 characters containing characters A through Z is added to the header.

For a description of using random values, see the subsequent subsections.

Random Strings

The device can generate random strings in header manipulation rules that may be substituted where the type ‘String’ is required. The random string can include up to 298 characters and include a range of, for example, from a to z or 1 to 10. This string is used in the table's 'Action Value' field.

The syntax for using random strings is:

```
Rand.string.<number of characters in string>.<low character>.<high
character>
```

Examples:

Rand.string.5.a.z: This generates a 5-character string using characters a through z.

Rand.string.8.0.z: This generates an 8-character string using characters and digits.

Random Integers

The device can generate a random numeric value that may be substituted where the type ‘Int’ is required. The syntax for random numeric values is:

```
Rand.number.<low number>.<high number>
```

Examples:

Rand.number.5.32: This generates an integer between 5 and 32

Wildcarding for Header Removal

The device supports the use of the "*" wildcard character to remove headers. The "*" character may only appear at the end of a string. For example, "X-*" is a valid wildcard request, but "X-*ID" is not.

Below are examples of using the wildcard:

header.p-* - removes all headers that have the prefix "p-"

header.via* - removes all Via headers

header.x-vendor* - removes all headers that start with "x-vendor"

header.* - removes all non-critical headers

header.to* - removes all headers that start with "to", except the To header, which is protected

Note: The wildcard does not remove the following headers: Request-Uri, Via, From, To, Callid, Cseq, and Contact.

Message Manipulation using SDP Conditions

You can configure message manipulation rules based on user-defined SDP conditions. For example, you can configure a manipulation rule to add a Diversion header to incoming INVITE messages if the SDP contains a specific IP address, or a prefix or suffix of this IP address.

The device supports the following SDP condition syntax:

param.message.sdp.address suffix '10.10'

param.message.sdp.address prefix '10.132'

param.message.sdp.address=='10.33.37.78'

The example below shows a manipulation rule that adds a Diversion header ("Diversion: <sip:12345@p4.isp.com>;reason=no-answer") to incoming INVITE messages, if the SDP contains the IP address 10.33.37.78 or the prefix of this IP address, i.e., 10.33. The IP address is contained in the "c=" line of the SDP (e.g., "c=IN IP4 10.33.37.75").

Figure B-1: Message Manipulation using SDP Conditions

The table below shows the example configuration:

Parameter	Rule Index 1	Rule Index 2
Message Type	invite	invite
Condition	param.message.sdp.address =='10.33.37.78'	param.message.sdp.address=='10.33'
Action Subject	header.diversion	header.diversion
Action Type	Add	Add
Action Value	<sip:12345@p4.isp.com>;reason=no-answer	<sip:12345@p4.isp.com>;reason=no-answer

You can configure several such manipulation rules and then apply them per IP Group using the 'Inbound Message Manipulation Set' parameter.

Copying Information between Messages using Variables

You can use variables in SIP message manipulation rules to copy specific information from one message to another. Information from one message is copied to a variable and then information from that variable is copied to any subsequent message. The device can store information in local or global variables. Local variables are stored on a per call basis and change when a new call is made. Up to two local variables can be used per call. Global variables do not change as new calls are made. Up to 10 global variables can be used.

The syntax for using variables is as follows:

Var.call.<src || dst><local index>

where *local index* is an integer between 1 and 2 inclusive

Var.global.<global index>

where *global index* is an integer between 1 and 10 inclusive

To store data in a variable, add the name of the variable in the Action Subject field and set the Action Type to Modify. To retrieve data from a variable, add it in the Action Value field and it can be used in any manipulation where a ManStringElement is valid as an Action Subject.

Below are examples of manipulation rules implementing variables:

Example 1:

Store a value in a call variable: Stores the subject URI parameter from the To header:

```
MessageManipulations 0 = 0, Invite.Request, , var.call.dst.1,
2, header.to.url.param.subject, 0;
```

Use the stored value: Allocates a Subject header for the 200 OK response for the same call and assigns it the stored value:

```
MessageManipulations 0 = 0, Invite.response.200, ,
header.subject, 0, var.call.dst.1, 0;
```

Example 2:

Store a value in a global variable: Stores the Priority header of the INVITE with ‘company’ in the host part of the From header:

```
MessageManipulations 0 = 0, Invite.Request,
header.from.url.host == ‘company’, var.global.1, 2,
header.priority, 0;
```

Use the stored value: Assigns the same priority as the INVITE request to SUBSCRIBE requests arriving with 'company' in the host part of the From header:

```
MessageManipulations 0 = 0, Subscribe.request,
header.from.url.host == ‘company’, header.priority, 0,
var.global.1, 0;
```

Enum Definitions

AgentRole

These ENUMs are applicable to the Server or User-Agent headers (see 'Server or User-Agent' on page 381).

Table B-9: Enum Agent Role

AgentRole	Value
Client	1
Server	2

Event Package

These ENUMs are applicable to the Server or User-Agent (see 'Server or User-Agent' on page 381) and Event (see 'Event' on page 366) headers.

Table B-10: Enum Event Package

Package	Value
TELEPHONY	1
REFER	2
REFRESH	3
LINE_STATUS	4
MESSAGE_SUMMARY	5
RTCPXR	6
SOFT_SYNC	7
CHECK_SYNC	8
PSTN	9
DIALOG_PACKAGE	10
REGISTRATION	11
START_CWT	12
STOP_CWT	13
UA_PROFILE	14
LINE_SEIZE	15

MLPP Reason Type

These ENUMs are applicable to the MLPP Structure (see 'MLPP' on page 389).

Table B-11: Enum MLPP Reason Type

Type	Value
PreEmption Reason	0
MLPP Reason	1

Number Plan

These ENUMs are applicable to the Remote-Party-Id header (see 'Remote-Party-Id' on page 377).

Table B-12: Enum Number Plan

Plan	Value
ISDN	1
Data	3
Telex	4
National	8
Private	9
Reserved	15

NumberType

These ENUMs are applicable to the Remote-Party-Id header (see 'Remote-Party-Id' on page 377).

Table B-13: Enum Number Type

Number Type	Value
INTERNATIONAL LEVEL2 REGIONAL	1
NATIONAL LEVEL1 REGIONAL	2
NETWORK PISN SPECIFIC NUMBER	3
SUBSCRIBE LOCAL	4
ABBREVIATED	6
RESERVED EXTENSION	7

Privacy

These ENUMs are applicable to the Remote-Party-Id (see 'Remote-Party-Id' on page 377) and Diversion (see 'Diversion' on page 365) headers.

Table B-14: Enum Privacy

Privacy Role	Value
Full	1
Off	2

Reason (Diversion)

These ENUMs are applicable to the Diversion header (see 'Diversion' on page 365).

Table B-15: Enum Reason

Reason	Value
Busy	1
No Answer	2
Unconditional	3
Deflection	4
Unavailable	5
No Reason	6
Out of service	7

Reason (Reason Structure)

These ENUMs are used in the Reason Structure (see 'Reason Structure' on page 390).

Table B-16: Enum Reason (Reason Structure)

Reason	Value
INVITE	5
REINVITE	6
BYE	7
OPTIONS	8
ACK	9
CANCEL	10
REGISTER	11
INFO	12
MESSAGE	13
NOTIFY	14
REFER	15
SUBSCRIBE	16

Reason	Value
PRACK	17
UPDATE	18
PUBLISH	19
LAST_REQUEST	20
TRYING_100	100
RINGING_180	180
CALL_FORWARD_181	181
QUEUED_182	182
SESSION_PROGRESS_183	183
OK_200	200
ACCEPTED_202	202
MULTIPLE_CHOICE_300	300
MOVED_PERMANENTLY_301	301
MOVED_TEMPORARILY_302	302
SEE_OTHER_303	303
USE_PROXY_305	305
ALTERNATIVE_SERVICE_380	380
BAD_REQUEST_400	400
UNAUTHORIZED_401	401
PAYMENT_REQUIRED_402	402
FORBIDDEN_403	403
NOT_FOUND_404	404
METHOD_NOT_ALLOWED_405	405
NOT_ACCEPTABLE_406	406
AUTHENTICATION_REQUIRED_407	407
REQUEST_TIMEOUT_408	408
CONFLICT_409	409
GONE_410	410
LENGTH_REQUIRED_411	411
CONDITIONAL_REQUEST_FAILED_412	412
REQUEST_TOO_LARGE_413	413
REQUEST_URI_TOO_LONG_414	414
UNSUPPORTED_MEDIA_415	415
UNSUPPORTED_URI_SCHEME_416	416
UNKNOWN_RESOURCE_PRIORITY_417	417
BAD_EXTENSION_420	420
EXTENSION_REQUIRED_421	421

Reason	Value
SESSION_INTERVAL_TOO_SMALL_422	422
SESSION_INTERVAL_TOO_SMALL_423	423
ANONYMITY_DISALLOWED_433	433
UNAVAILABLE_480	480
TRANSACTION_NOT_EXIST_481	481
LOOP_DETECTED_482	482
TOO_MANY_HOPS_483	483
ADDRESS_INCOMPLETE_484	484
AMBIGUOUS_485	485
BUSY_486	486
REQUEST_TERMINATED_487	
NOT_ACCEPTABLE_HERE_488	488
BAD_EVENT_489	489
REQUEST_PENDING_491	491
UNDECIPHERABLE_493	493
SECURITY_AGREEMENT_NEEDED_494	494
SERVER_INTERNAL_ERROR_500	500
NOT_IMPLEMENTED_501	501
BAD_GATEWAY_502	502
SERVICE_UNAVAILABLE_503	503
SERVER_TIME_OUT_504	504
VERSION_NOT_SUPPORTED_505	505
MESSAGE_TOO_LARGE_513	513
PRECONDITION_FAILURE_580	580
BUSY_EVERYWHERE_600	600
DECLINE_603	603
DOES_NOT_EXIST_ANYWHERE_604	604
NOT_ACCEPTABLE_606	606

Reason (Remote-Party-Id)

These ENUMs are applicable to the Remote-Party-Id header (see 'Remote-Party-Id' on page 377).

Table B-17: Enum Reason (RPI)

Reason	Value
Busy	1
Immediate	2
No Answer	3

Refresher

These ENUMs are used in the Session-Expires header (see 'Session-Expires' on page 383).

Table B-18: Enum Refresher

Refresher String	Value
UAC	1
UAS	2

Screen

These ENUMs are applicable to the Remote-Party-Id (see 'Remote-Party-Id' on page 377) and Diversion (see 'Diversion' on page 365) headers.

Table B-19: Enum Screen

Screen	Value
Yes	1
No	2

ScreenInd

These ENUMs are applicable to the Remote-Party-Id header (see 'Remote-Party-Id' on page 377).

Table B-20: Enum ScreenInd

Screen	Value
User Provided	0
User Passed	1
User Failed	2
Network Provided	3

TransportType

These ENUMs are applicable to the URL Structure (see 'URL' on page 391) and the Via header (see 'Via' on page 386).

Table B-21: Enum TransportType

TransportType	Value
UDP	0
TCP	1
TLS	2
SCTP	3

Type

These ENUMs are applicable to the URL Structure (see 'URL' on page 391).

Table B-22: Enum Type

Type	Value
SIP	1
Tel	2
Fax	3
SIPS	4

Actions and Types

Table 34-23: Action and Types

Element Type	Command Type	Command	Value Type	Remarks
IPGroup	Match	==	String	Returns true if the parameter equals to the value.
		!=	String	Returns true if the parameter not equals to the value.
		contains	String	Returns true if the string given is found in the parameter value.
		!contains	String	Returns true if the string given is not found in the parameter value.
Call-Parameter	Match	==	String	Returns true if the parameter equals to the value.
		!=	String	Returns true if the parameter not equals to the value.
		contains	String	Returns true if the string given is found in the parameter value.
		!contains	String	Returns true if the string given is not found in the parameter value.
Body	Match	==	String	Returns true if the body's content equals to the value.
		!=	String	Returns true if the body's content not equals to the value.
		contains	String	Returns true if the string given is found in the body's content.
		!contains	String	Returns true if the string given is not found in the body's content.
		exists		Returns true if this body type exists in the message.
		!exists		Returns true if this body type does not exist in the message.
	Action	Modify	String	Modifies the body content to the new value.
		Add	String	Adds a new body to the message. If such body exists the body content will be modified.
		Remove		Removes the body type from the message.
Header-List	Match	==	String *Header-list	Returns true if the header's list equals to the string.
		!=	String *Header-list	Returns true if the header's list not equals to the string.
		contains	String	Returns true if the header's list contains the

Element Type	Command Type	Command	Value Type	Remarks
				string.
		!contains	String	Returns true if the header's list does not contain the string.
		exists		Returns true if at least one header exists in the list.
		!exists		Returns true if no headers exist in the list.
	Action	Modify	String *Header	Removes all the headers from the list and allocates a new header with the given value.
		Add	String *Header	Adds a new header to the end of the list.
		Remove		Removes the whole list from the message.
Header	Match	==	String *Header	Returns true if a header equals to the value. The header element must not be a list.
		!=	String *Header	Returns true if a header not equals to the value. The header element must not be a list.
		contains	String	Returns true if the header contains the string.
		!contains	String	Returns true if the header does not contain the string.
		exists		Returns true if the header exists.
		!exists		Returns true if the header does not exist.
	Action	Modify	String *Header	Replaces the entire header with the new value.
		Remove		Removes the header from the message, if the header is part of a list only that header will be removed.
		Add	String *Header	Adds a new header to the end of the list.
Parameter -List	Match	==	String Parameter-list	Returns true if the header's list equals to the string.
		!=	String Parameter-list	Returns true if the header's list not equals to the string.
		contains	String	Returns true if the header's list contains the string.
		!contains	String	Returns true if the header's list does not

Element Type	Command Type	Command	Value Type	Remarks
				contain the string.
		exists		Returns true if at least one parameter exists in the list.
		!exists		Returns true if the header's parameter list is empty.
	Action	Modify	String Parameter-list	Replaces the current parameters with the new value.
		Add	String Parameter	Adds a new parameter to the parameter's list.
		Remove		Removes all the unknown parameters from the list.
Parameter	Match	==	String Parameter	Returns true if the header's parameter's value equals to the value.
		!=	String Parameter	Returns true if the header's parameter's value not equals to the value.
		contains	String	Returns true if the header's parameter contains the string.
		!contains	String	Returns true if the header's parameter does not contain the string.
		exists		Returns true if the header's parameter exists.
		!exists		Returns true if the header's parameter does not exist.
	Action	Modify	String Parameter	Sets the header's parameter to the value.
		Remove		Removes the header's parameter from the parameter list.
Structure	Match	==	String *Structure	Returns true if the header's structure's value equals to the value. The string given must be able to be parsed to the structure.
		!=	String *Structure	Returns true if the header's structure's value not equals to the value. The string given must be able to be parsed to the structure.
	Action	Modify	String *Structure	Sets the header's structure to the value. The string given must be able to be parsed to the structure.
Integer	Match	==	Integer	Returns true if value equals to the integer element
		!=	Integer	Returns true if value not equals to the integer element

Element Type	Command Type	Command	Value Type	Remarks
		>	Integer	Returns true if value is greater than the value.
		>=	Integer	Returns true if value is greater than or equals to the value.
		<	Integer	Returns true if value is less than the value.
		<=	Integer	Returns true if value is less than or equals to the value.
	Action	Modify	Integer	Sets the integer element to the value. A string value must be a representation of an integer.
String	Match	==	String	Returns true if the string element equals to the value.
		!=	String	Returns true if the string element not equals to the value.
		contains	String	Returns true if the value is found in the string element.
		!contains	String	Returns true if the value is not found in the string element.
		>	String	Performs a character by character compare. Returns true if the ASCII value of the character is greater than that in the value
		>=	String	Performs a character by character compare. Returns true if the ASCII value of the character is greater than or equal to that in the value
		<	String	Performs a character by character compare. Returns true if the ASCII value of the character is less than that in the value
		<=	String	Performs a character by character compare. Returns true if the ASCII value of the character is less than or equal to that in the value
	Action	Modify	String	Sets the string element to the value.
		Add prefix	String	Adds the value to the beginning of the string element.
		Remove prefix	String	Removes the value from the beginning of the string element.
		Add suffix	String	Adds the value to the end of the string element.
		Remove suffix	String	Removes the value from the end of the string element.
Boolean	Match	==	Boolean	Returns true if the Boolean element equals to the value. Boolean – can be either 0 or 1.

Element Type	Command Type	Command	Value Type	Remarks
		!=	Boolean	Returns true if the Boolean element not equals to the value. Boolean – can be either 0 or 1.
		>	Boolean	Returns true if the Boolean element not equals to the value. Boolean – can be either 0 or 1.
		<	Boolean	Returns true if the Boolean element not equals to the value. Boolean – can be either 0 or 1.
	Action	Modify	Boolean	Sets the Boolean element to the value. Boolean – can be either 0 or 1.
Attribute	Match	==	Integer *Attribute	Returns true if the attribute element equals to the value. An attribute element value must be of the same type of the attribute element.
		!=	Integer *Attribute	Returns true if the attribute element not equals to the value. An attribute element value must be of the same type of the attribute element.
	Action	Modify	Integer *Attribute	Sets the attribute element to the value. An attribute element value must be of the same type of the attribute element.

Syntax

Rules table:

Man Set ID	Message Type	Condition	Action Element	Action Type	Action Value	Row Rule
ID	<message-type>	<match-condition>	<message-element>	<action-type>	<value>	ID

2. **message-type:**

Description: Rule is applied only if this is the message's type

Syntax:<method>.<message-role>

Examples:

invite.request

invite.response.200

subscribe.response.2xx

a. **method:**

Description: Rule is applied only if this is the message's method

Syntax: token / any

Examples:

Invite, subscribe	rule applies only to INVITE messages
Unknown	unknown methods are also allowed
Any	no limitation on the method type

message-role

Description: Rule is applied only if this is the message's role

Syntax: request / response.response-code / any

Examples:

Request	rule applies only on requests
Response.200	rule applies only on 200 OK messages
Any	no limitations on the type of the message

response-code

Description: Response code of the message

Syntax: 1xx / 2xx / 3xx / 4xx / 5xx / 6xx / 3digit / any

Examples:

3xx	any redirection response
200	only 200 OK response
Any	any response

match-condition:

Description: Matching criteria for the rule

Syntax: (message-element / param) SWS match-type [SWS value] * [SWS logical-expression SWS match-condition]

Examples:

header.from.user == '100'

header.contact.header-param.expires > '3600'

header.to.host contains 'itsp'

param.call.dst.user != '100'

header.john exists

header.john exists AND header.to.host !contains 'john'

header.from.user == '100' OR header.from.user == '102' OR header.from.user == '300'

a. match-type

Description: Comparison to be made

Syntax:

==	equals
!=	not equals
>	greater than
<	less than
>=	greater than or equal to
<=	less than or equal to
contains	does a string contain a value (relevant only to string fields)
exists	does a certain header exists
!exists	does a certain header not exists
!contains	does a string exclude a value. Relevant only to string fields

logical-expression:

Description: Condition for the logical expression

Syntax:

ANDlogical And

OR logical Or

Note: "A AND B OR C" is calculated as A AND (B OR C).

message-element:

Description: Element in the message

Syntax: (header / body).message-element-name [.header-index] * [.(sub-element / sub-element-param)]

Examples:

header.from

header.via.2.host

header.contact.header-param.expires

header.to.uri-param.user-param

body.application/dtmf-relay

a. message-element-name

Description: Name of the message's element - "/" only used for body types

Syntax: 1 * (token / "/")

Examples:

from (header's name)

to (header's name)

application/dtmf-relay (body's name)

header-index

Description: Header's index in the list of headers

Syntax: Integer

Examples: If five Via headers arrive:

0 (default)	refers to first Via header in message
1	second Via header
4	fifth Via header

sub-element

Description: Header's element

Syntax: sub-element-name

Examples:

user

host

sub-element-param

Description: Header's element

Syntax: sub-element-name [.sub-element-param-name]

Example:

header.from.param.expires

sub-element-param-name

Description: Header's parameter name - relevant only to parameter sub-elements

Syntax: token

Examples:

expires (contact's header's param)

duration (retry-after header's param)

unknown-param (any unknown param can be added/removed from the header)

param

Description: Params can be as values for match and action

Syntax: param.param-sub-element.param-dir-element.(call-param-entity / ipg-param-entity)

Examples:

param.ipg. src.user

param.ipg.dst.host

param.ipg.src.type

param.call.src.user

param-sub-element

Description: Determines whether the param being accessed is a call or an IP Group

Syntax:

call	relates to source or destination URI for the call
ipg	relates to source or destination IP Group

param-dir-element

Description: Direction relating to the classification

Syntax:

src	refers to source
ds	refers to destination

call-param-entity

Description: Parameters that can be accessed on the call

Syntax:

user	refers to username in request-URI for call

ipg-param-entity

Description: Name of the parameter

Syntax:

user	refers to Contact user in IP Group
host	refers to Group Name in IP Group table
type	refers to Type field in IP Group table
id	refers to IP Group ID (used to identify source or destination IP Group)

string

Description: String

Syntax: stringenclosed in single apostrophe

Examples:

'username

'123'

'user@host'

integer

Description: A number

Syntax: 1 * digit

Example:

123

action-type:

Description: Action to be performed on the element

Syntax:

modify sets element to new value (all element types)

add-prefix adds value at beginning of string (string element only)

remove-prefix removes value from beginning of string (string element only)

add-suffix adds value at end of string (string element only)

remove-suffix removes value from end of string (string element only)

add adds a new header/param/body (header or parameter elements)

remove removes a header/param/body (header or parameter elements)

value:

Description: Value for action and match

Syntax: ('string' / message-element / param) * (+ ('string' / message-element / param))

Examples:

'itsp.com'

header.from.user

param.ipg.src.user

param.ipg.dst.host + '.com'

param.call.src.user + '<' + header.from.user + '@' + header.p-asserted-id.host + '>'

Technical Specifications

The device's technical specifications are listed in the table below.

Table C-1: Technical Specifications

Function	Specification
Networking Interfaces	
LAN	Two1000Base-T (Gigabit Ethernet) LAN port interfaces Physical port separation by selecting port group per network interface
High Availability (HA)	
Full HA	Two deployed devices for 1+1 high availability, communicating through a Maintenance network interface. Upon failure of the active device, all functionality is switched over to the redundant device.
Media Processing	
IP Transport	VoIP (RTP/RTCP) per IETF RFC 3550 and 3551, IPv6
Control and Management	
Control Protocols	SIP-TCP, UDP, TLS and MSCML Stand Alone Survivability for service continuity
Operations & Management	Embedded HTTP Web Server, Telnet, SNMP V2/V3 Remote configuration and software download via TFTP, HTTP, HTTPS, DHCP RADIUS, Syslog (for events, alarms and CDRs)
IP/VoIP Quality of Service	
	IEEE 802.1p, TOS, DiffServ IEEE 802.1Q VLAN tagging Shaping, Policing, Queuing, Bandwidth Reservation
Session Border Controller (SBC)	
	SIP Header conversion: IP to IP Routing translations of SIP, UDP, TCP, TLS Translation of RTP, SRTP; Support SIP trunk with multi-ITSP (Registrations to ITSPs is invoked independently); Topology hiding; Call Admission Control; Call Black/White list Intrusion detection/prevention (NIDS); Anti SPIT & SPAM mechanisms
Hardware Specifications	
Recommended Platform	Platform: HP ProLiant DL120 G7 Processor:Intel Xeon E3-1220 (8M Cache, 3.10 GHz), 4 Cores Memory: 8 GB Disk space:72 GB or more CD-ROM:Local CLI support: VGA monitor and keyboard RS-232 serial port (optional)